UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Parallelisation of stellar codes

David Martin

UPC Barcelona, Spain
Group of Astronomy and Astrophysics (GAA)
April 18th 2013

IMAGE SOURCE: Harvard-Smithsonian Center for Astrophysics

Outline

1. Introduction
2. Designing Parallel Applications
3. Parallelisation of a Nucleosynthesis Code
4. Parallelisation of a multi-zone Hydrodynamic Code
5. Conclusions

Introduction (I)

- Nucleosynthesis in stellar environments is investigated using computer models to evaluate the evolution of abundances according to the time-evolution of their reaction rates in an astrophysical system.

- Nucleosynthesis can sometimes be computed synchronously with the hydrodynamical system (time-consuming process).

- To alleviate this constraint, and depending on the number of nuclides needed, two different strategies can be adopted with regard to the nuclear network used:

  – (i) use a reduced nuclear network which is responsible for energy generation and is therefore essential for any accurate hydrodynamical evolution of the system.

  – (ii) use a reaction network that is necessary for detailed computation of nucleosynthesis (where **T** and **ρ** obtained from step (i) are used).

Introduction (II)

- Parallel computing has been raised as the main permitting factor of more precise, and computationally intensive simulations.

- Two different parallelization scenarios have been studied:

  – Parallelization of the nucleosynthesis portion of the code, by parallelizing the solution of ordinary differential equations for the nuclear reaction network. Nucleosynthesis is by far the most time-consuming part of the calculations.

  – Parallelization of the whole multi-zone hydrodynamic calculation. Performance results will be shown on the spherically symmetric, Lagrangian, hydrodynamic code SHIVA (José 1996; José & Hernanz 1998)

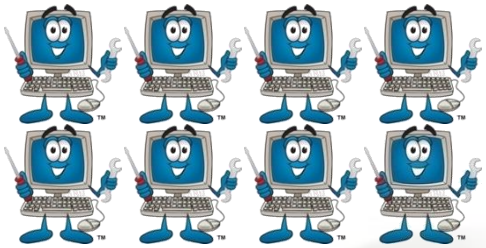
Where are we?

1. Introduction
2. Designing Parallel Applications
3. Parallelisation of a Nucleosynthesis Code
4. Parallelisation of a multi-zone Hydrodynamic Code
5. Conclusions

Parallel Computer

- A parallel computer is formed by a group of processors that execute specific tasks cooperatively with the goal of solving a particular computational problem.

- In principle, parallelism is as *simple* as applying N processors or CPUs to a single problem, expecting to solve it N times faster.



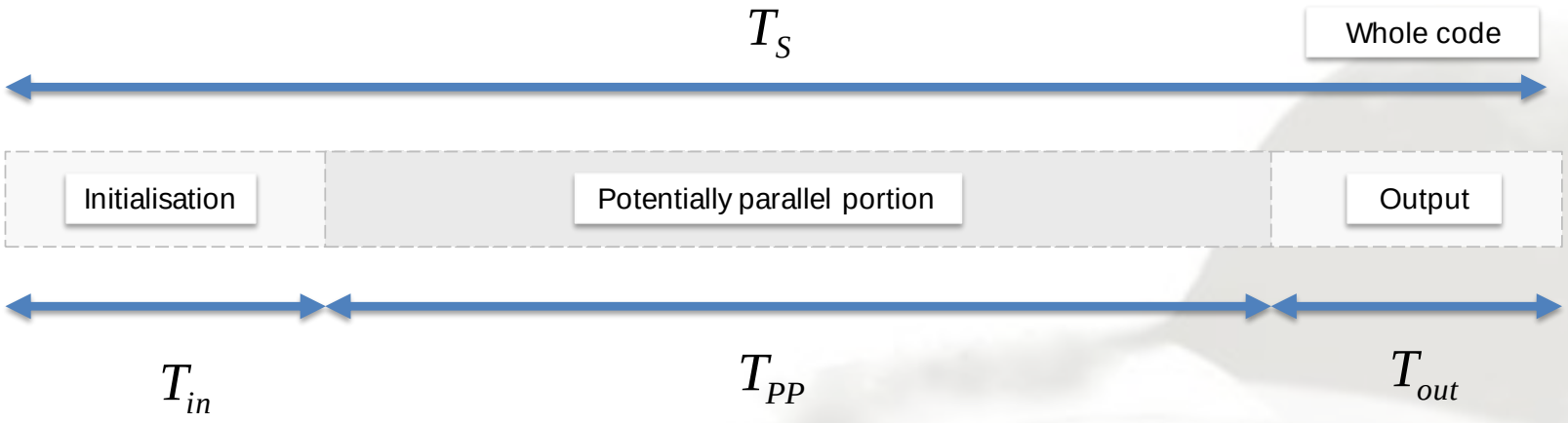
Source: toons4biz.com

- However, speed-ups hinge on the nature of the problem being parallelised: **not all applications can be parallelised effectively**.

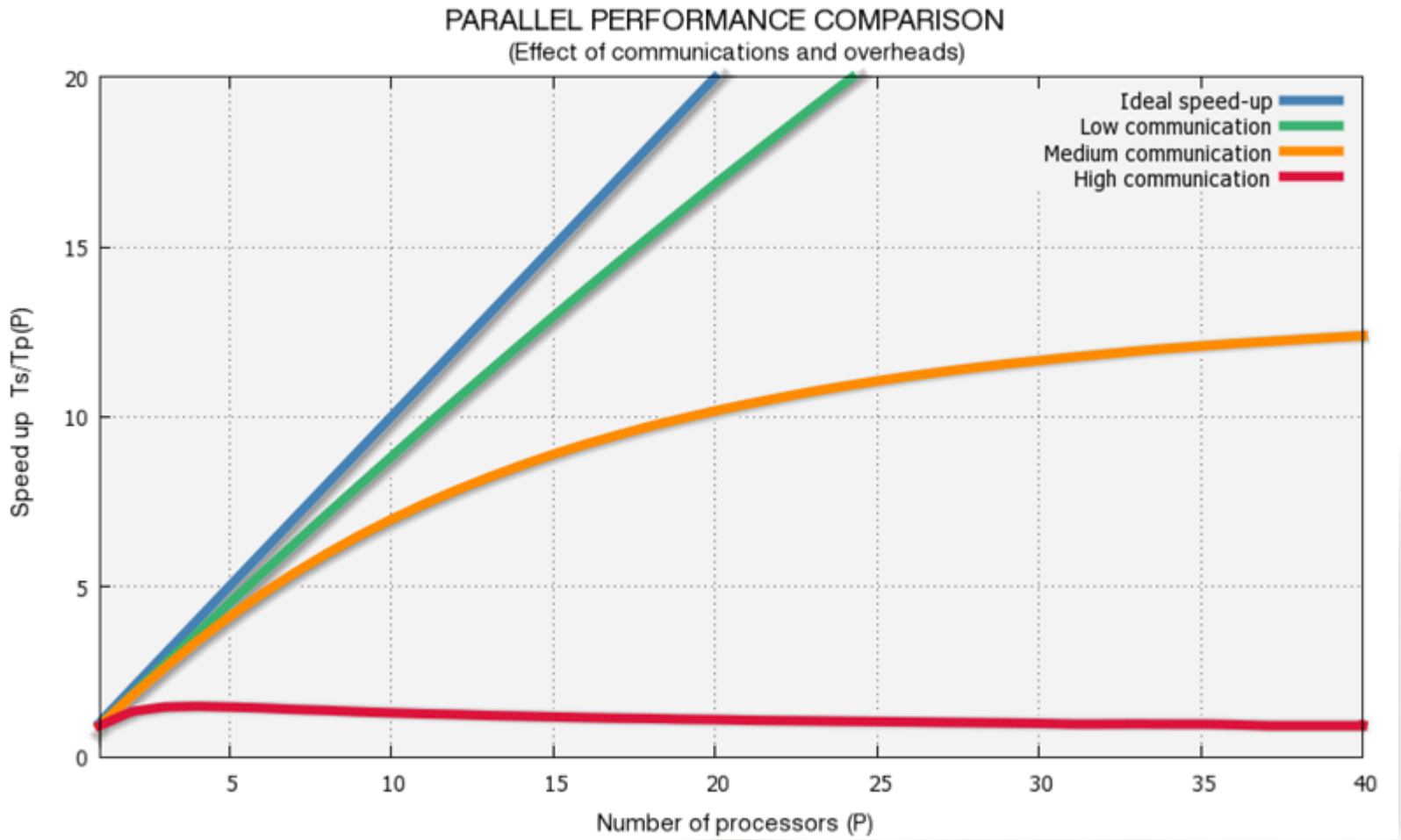
Speed-up Model

- The Speed-up accomplished with the parallelisation can be approximated by:

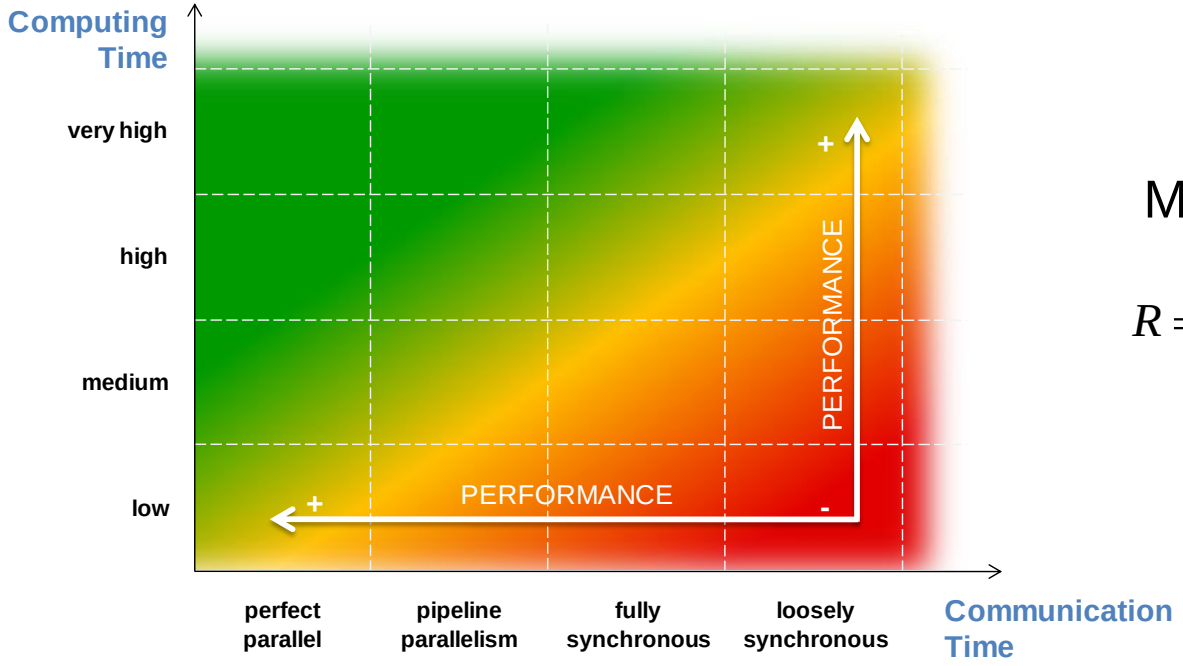
$$\text{Speed} - \text{up} \approx \frac{T_s}{T_{in} + \frac{T_{pp}}{N_p} + T_{comm} + T_{out}} = \frac{1}{(1-p) + \left(\frac{p}{N_p}\right) + \frac{T_{comm}}{T_S}}; \qquad p = \frac{T_{pp}}{T_S}$$



Parallel Performance



When is parallelisation effective?



Maximise the ratio:

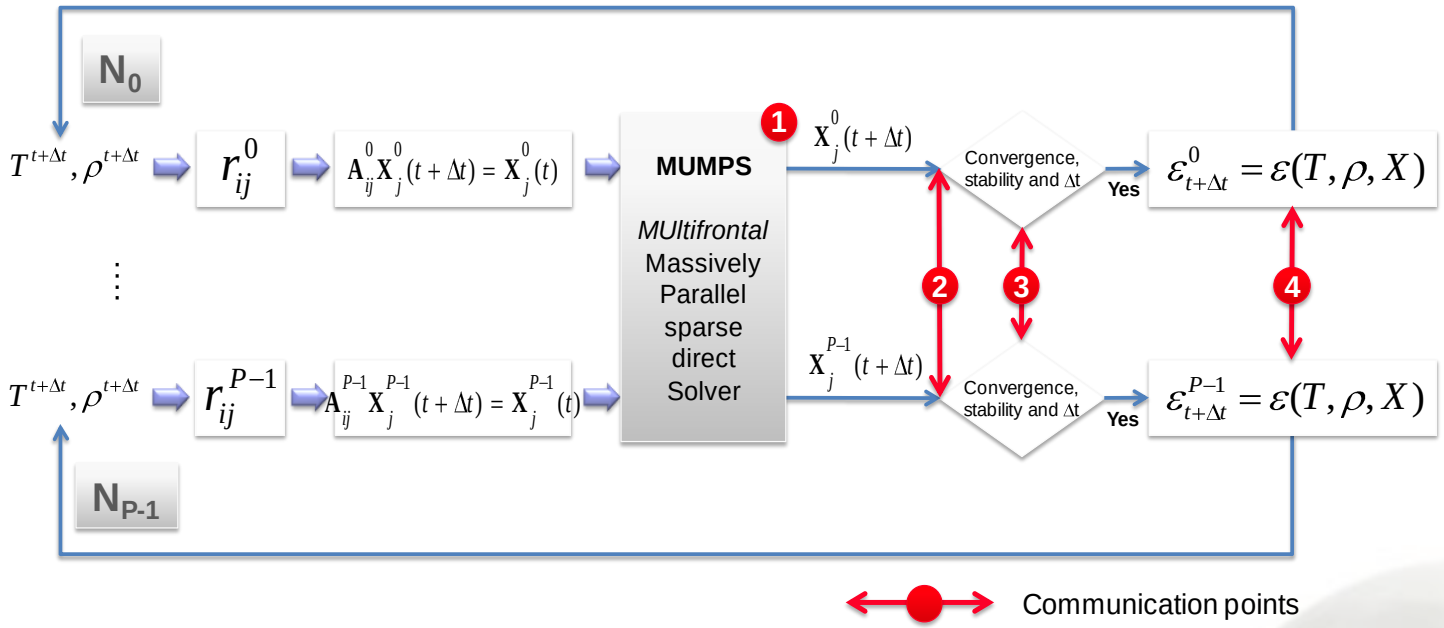
$$R = \frac{Computation\ time}{Communication\ time}$$

- To obtain good performance, communication time has to be kept to a small fraction of all computing time (Pancake et al. 1994).

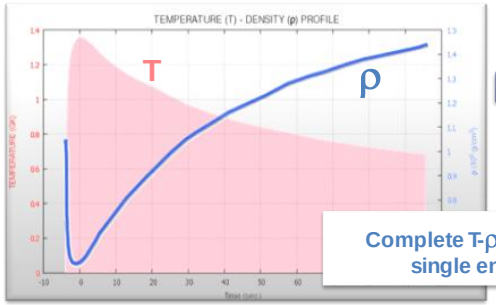
Where are we?

1. Introduction
2. Designing Parallel Applications
3. Parallelisation of a Nucleosynthesis Code
4. Parallelisation of a multi-zone Hydrodynamic Code
5. Conclusions

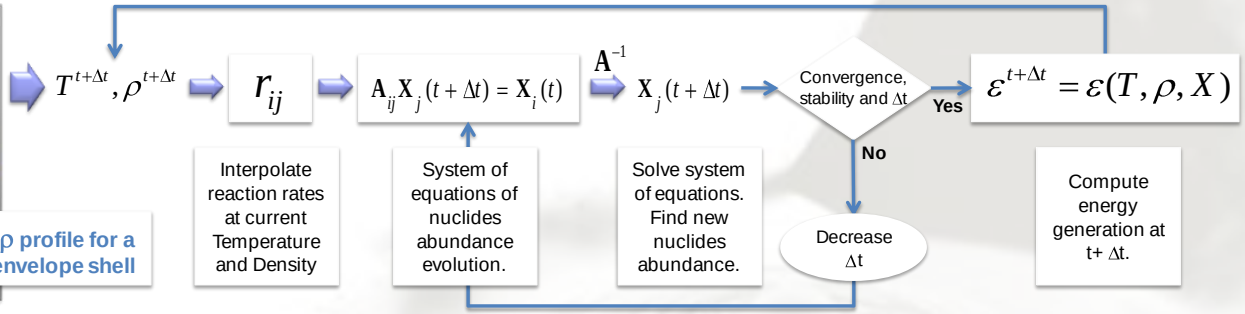
Post-processing parallelisation strategy



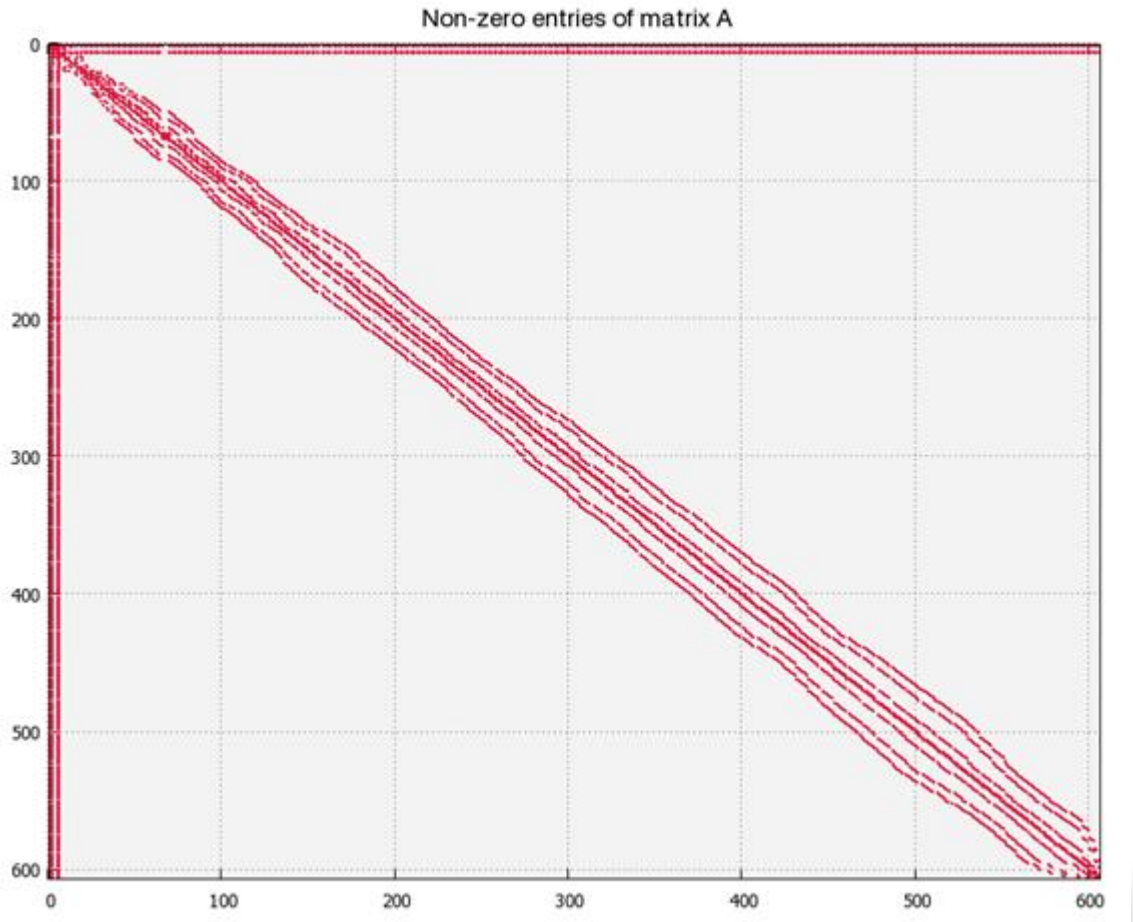
METHOD OF COMPUTATION: Post-processing nucleosynthesis code



Complete T-ρ profile for a single envelope shell



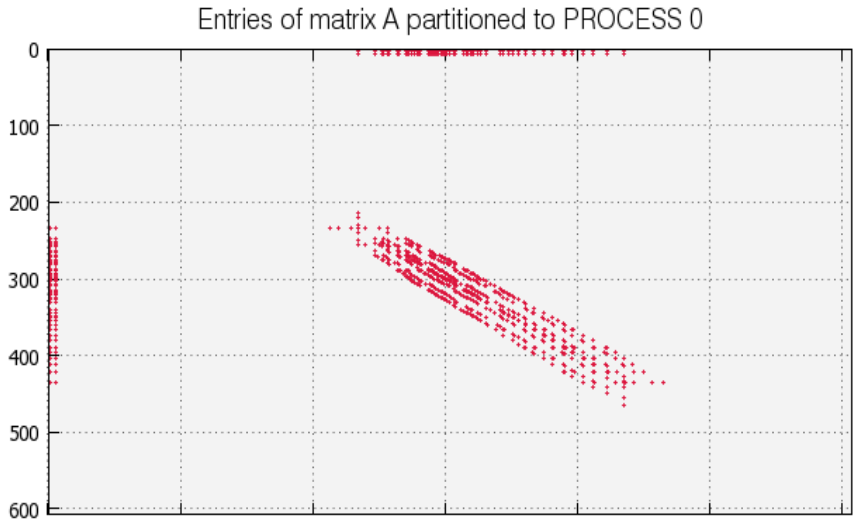
Parallel Matrix Assembly (I)

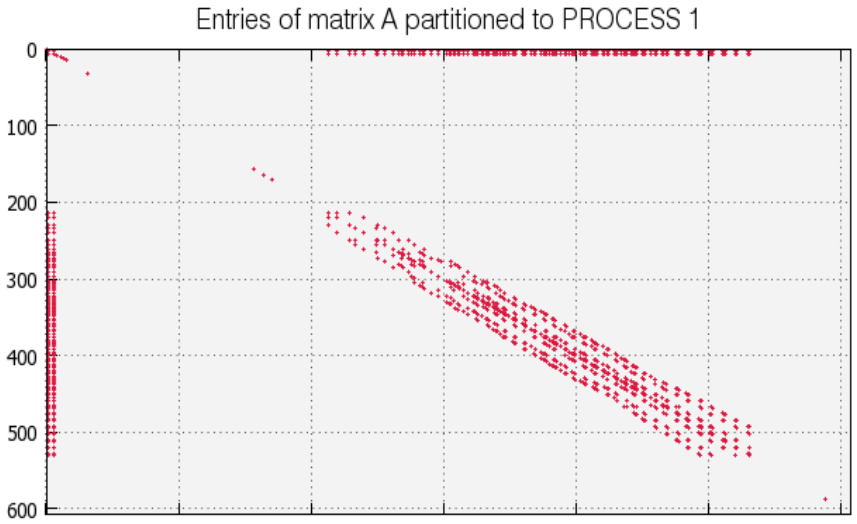


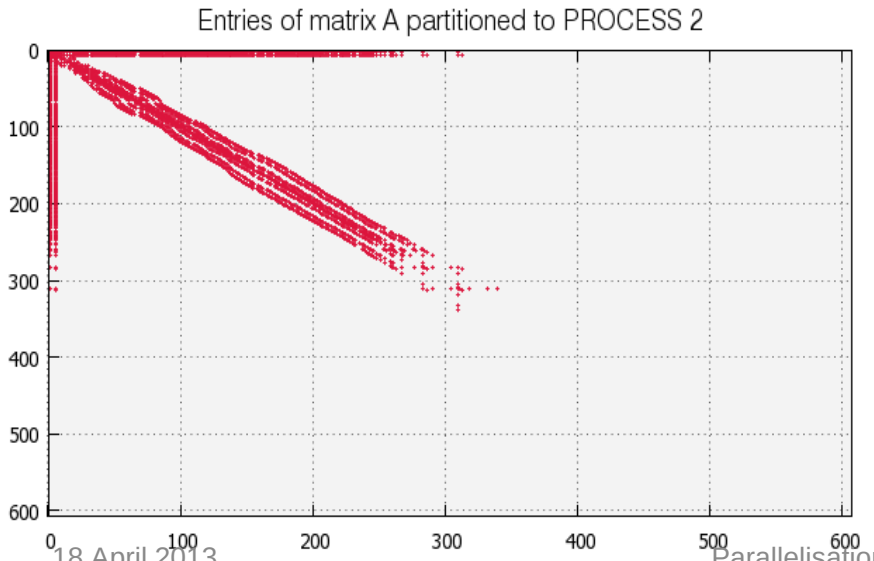
One of the most time consuming stages of the computation is the construction of the matrix **A** that arises from the linearisation of the set of differential equations describing the time evolution of the network abundances:

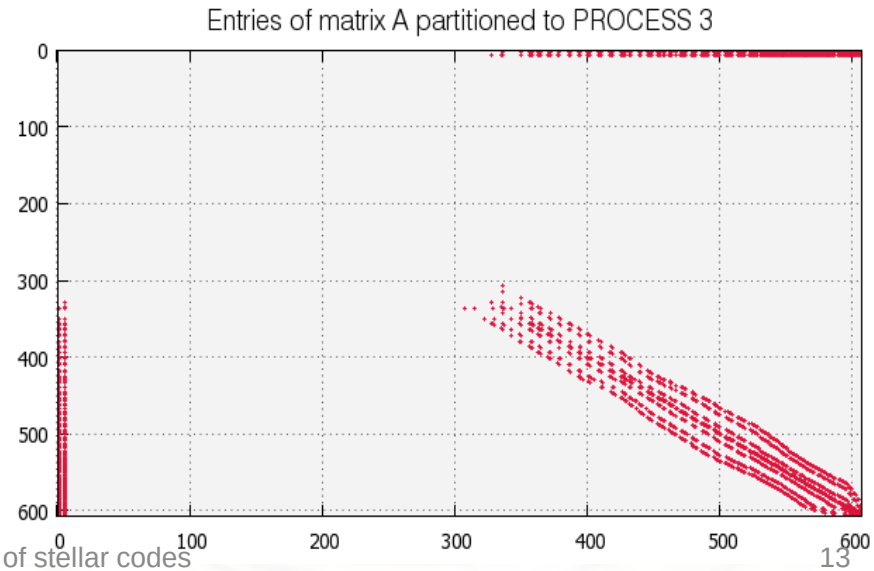
$$\mathbf{A} \cdot \mathbf{X} = \mathbf{X}_0$$

Parallel Matrix Assembly (II)

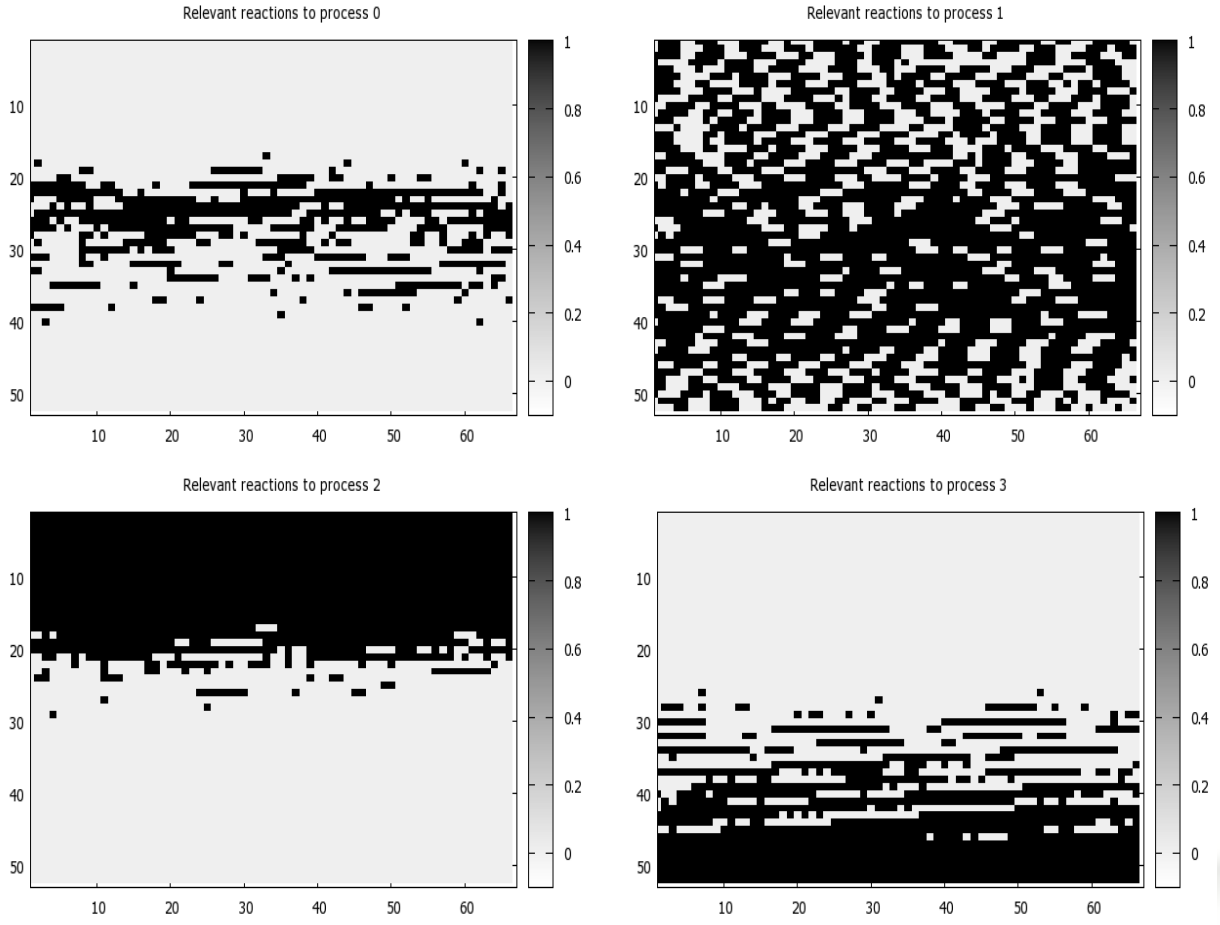








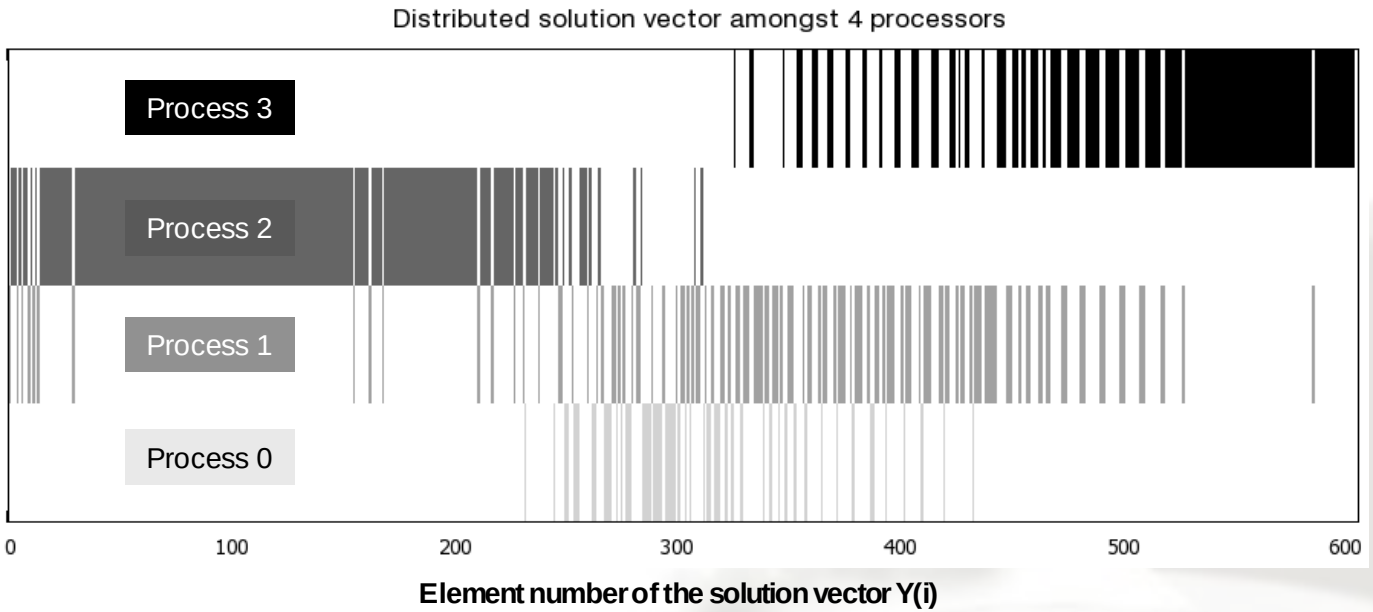
Interpolation of Reaction Rates



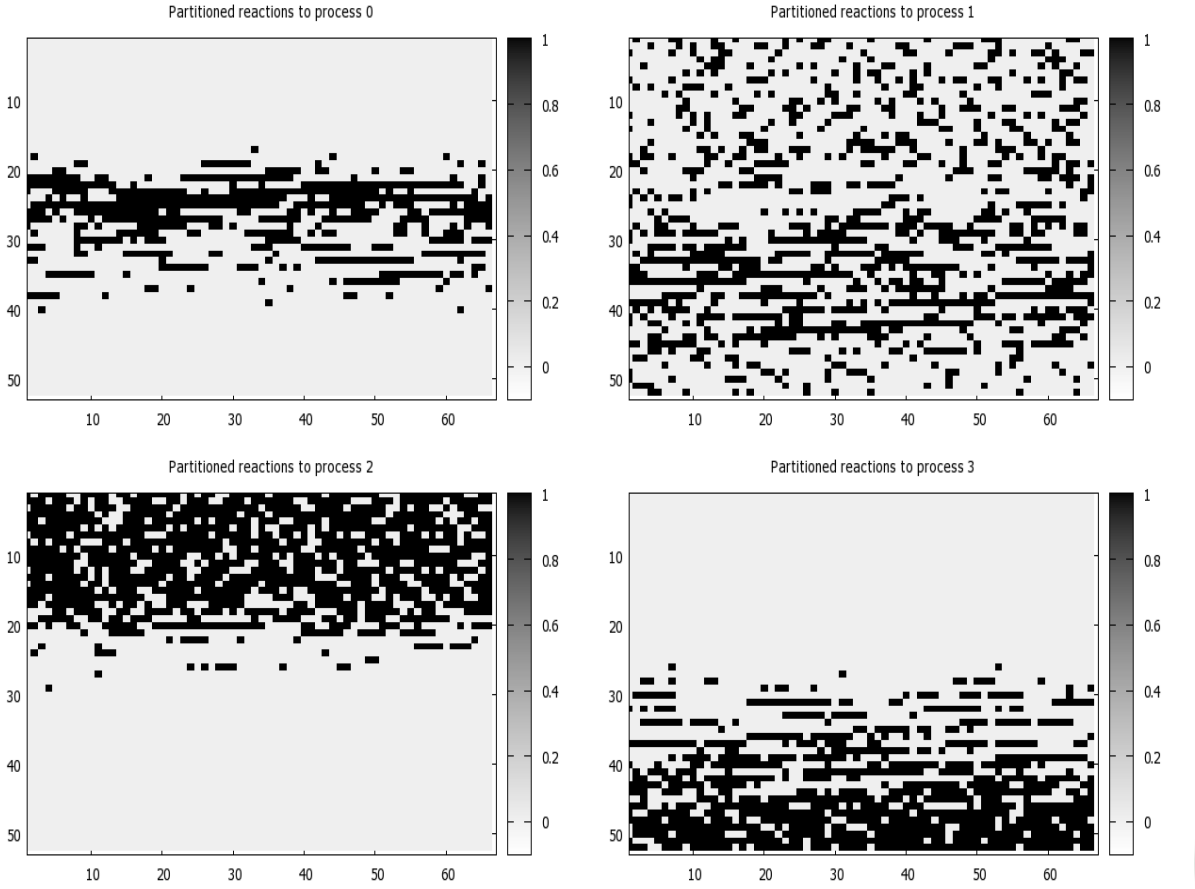
Each node performs the interpolation of those reaction rates that it will be using afterwards in the construction of their local partition of the matrix A.

Solution of the System of Equations

- The system of equations is solved using MUMPS; a software application for the solution of sparse systems of linear algebraic equations **Ax = b** on distributed memory parallel computers.
- Each node obtains a part of the solution vector.



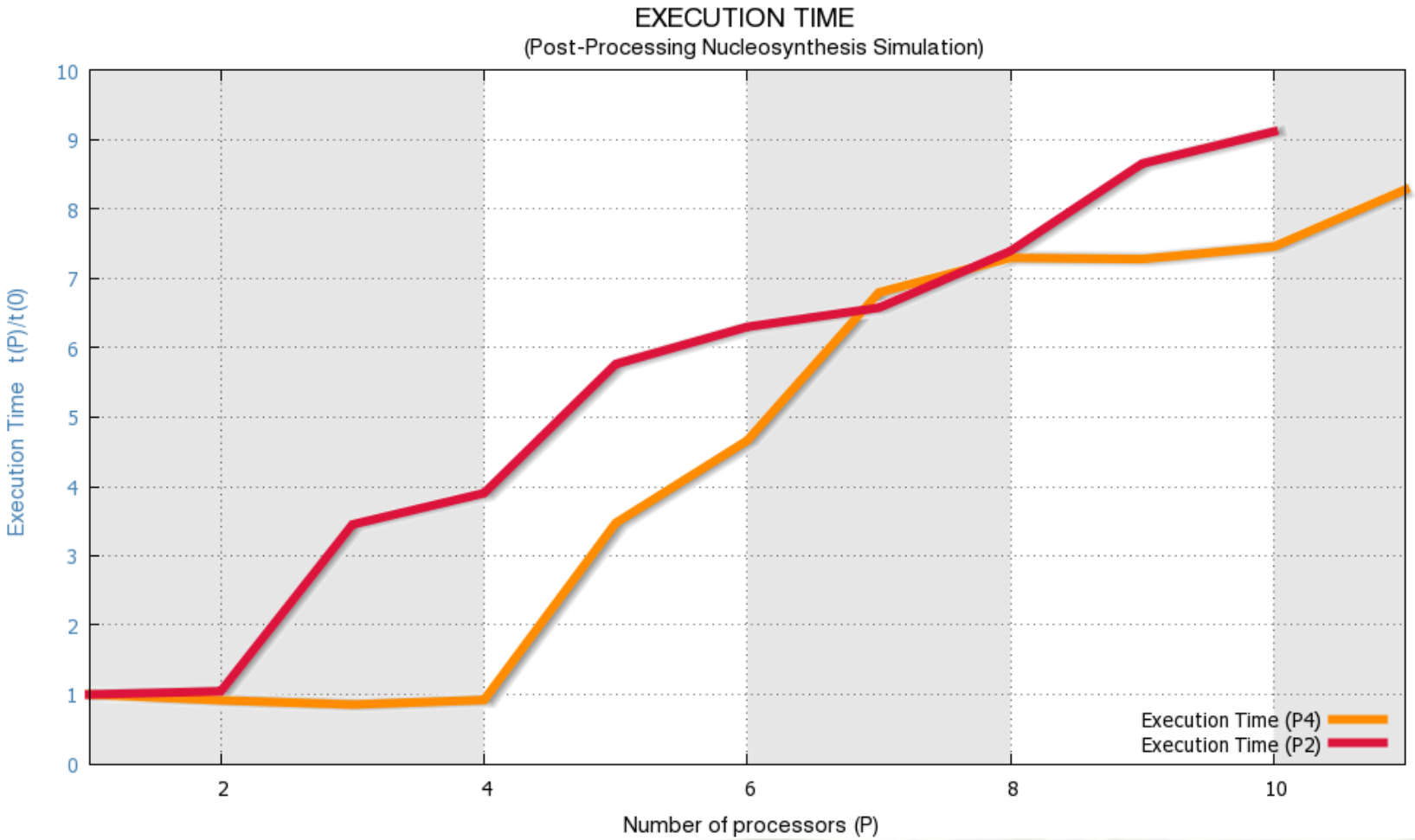
Energy Released Computation



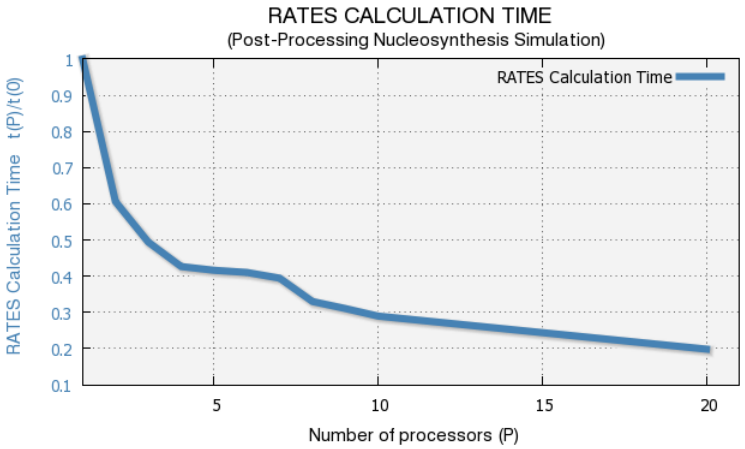
Each node performs the calculation of the Energy Released for specific nuclear reactions.

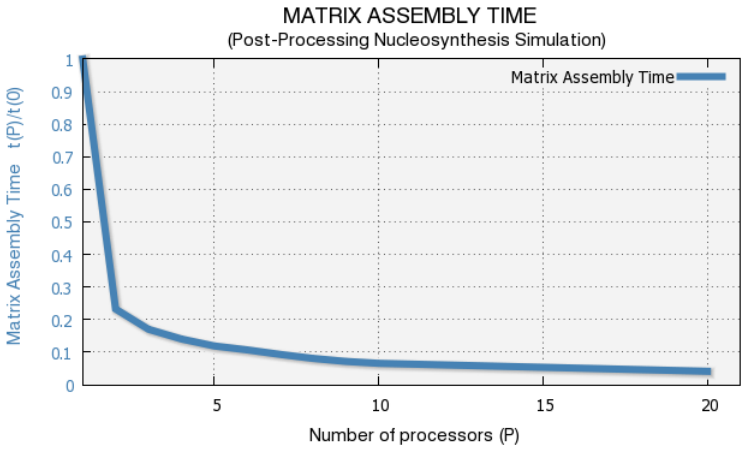
Afterwards, contributions from all nodes are summed up to account for the total released energy ε_{total}

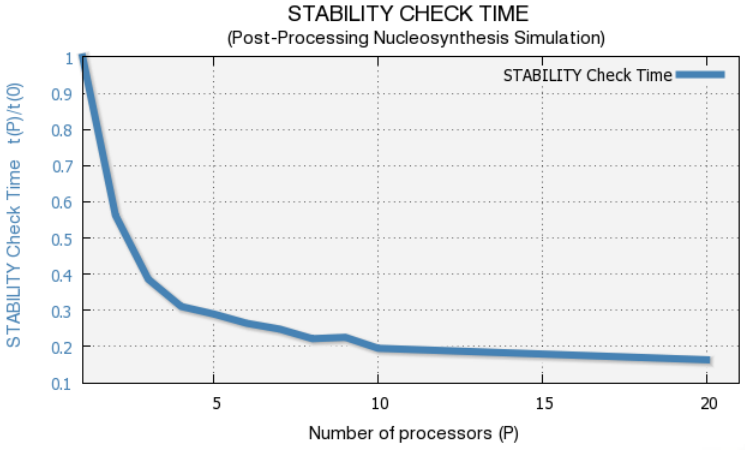
Performance: Total execution time

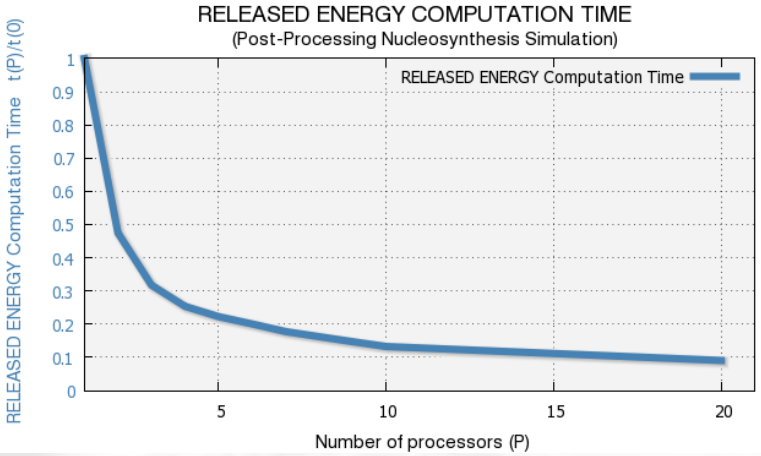


Performance: Partial executions time

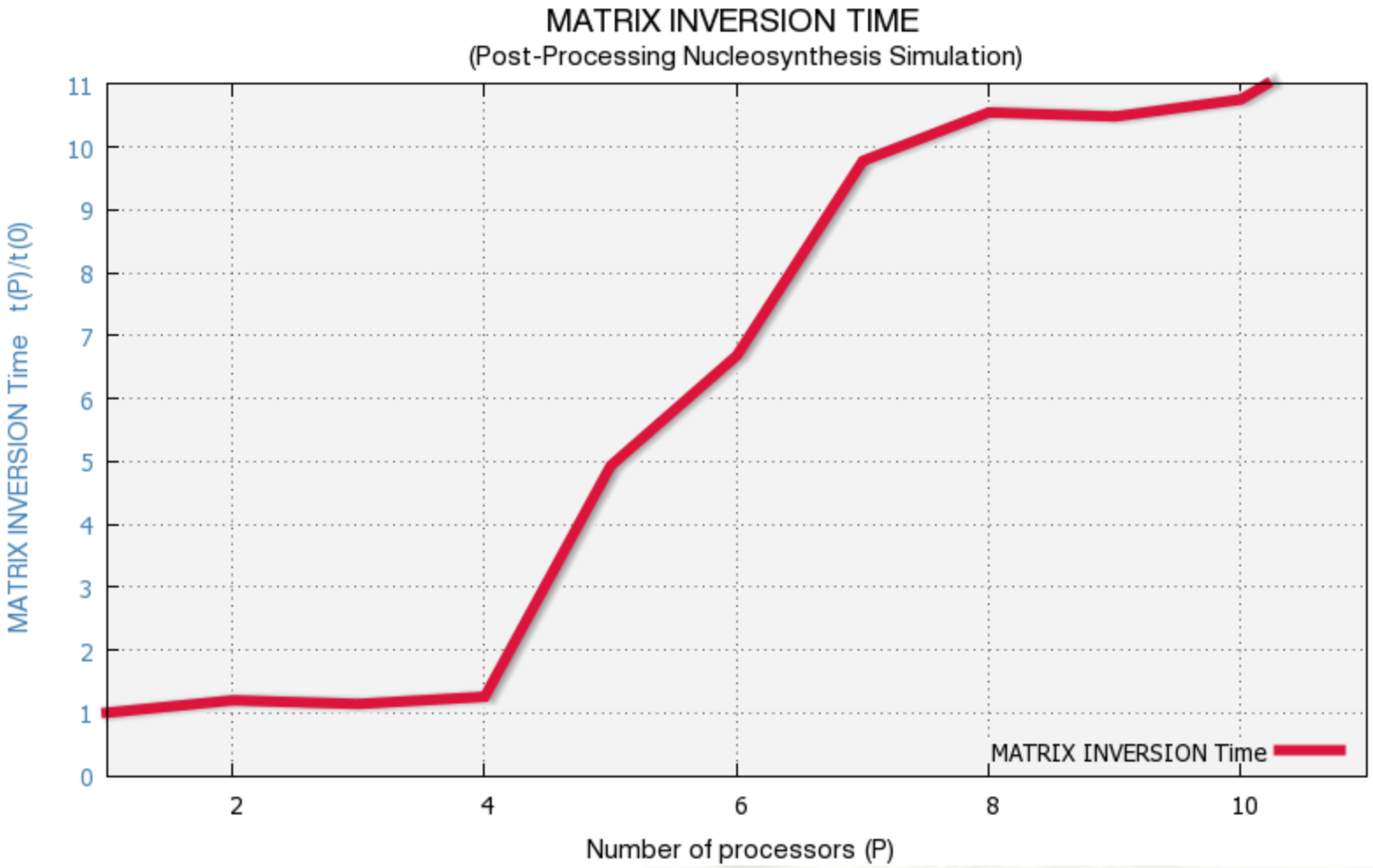




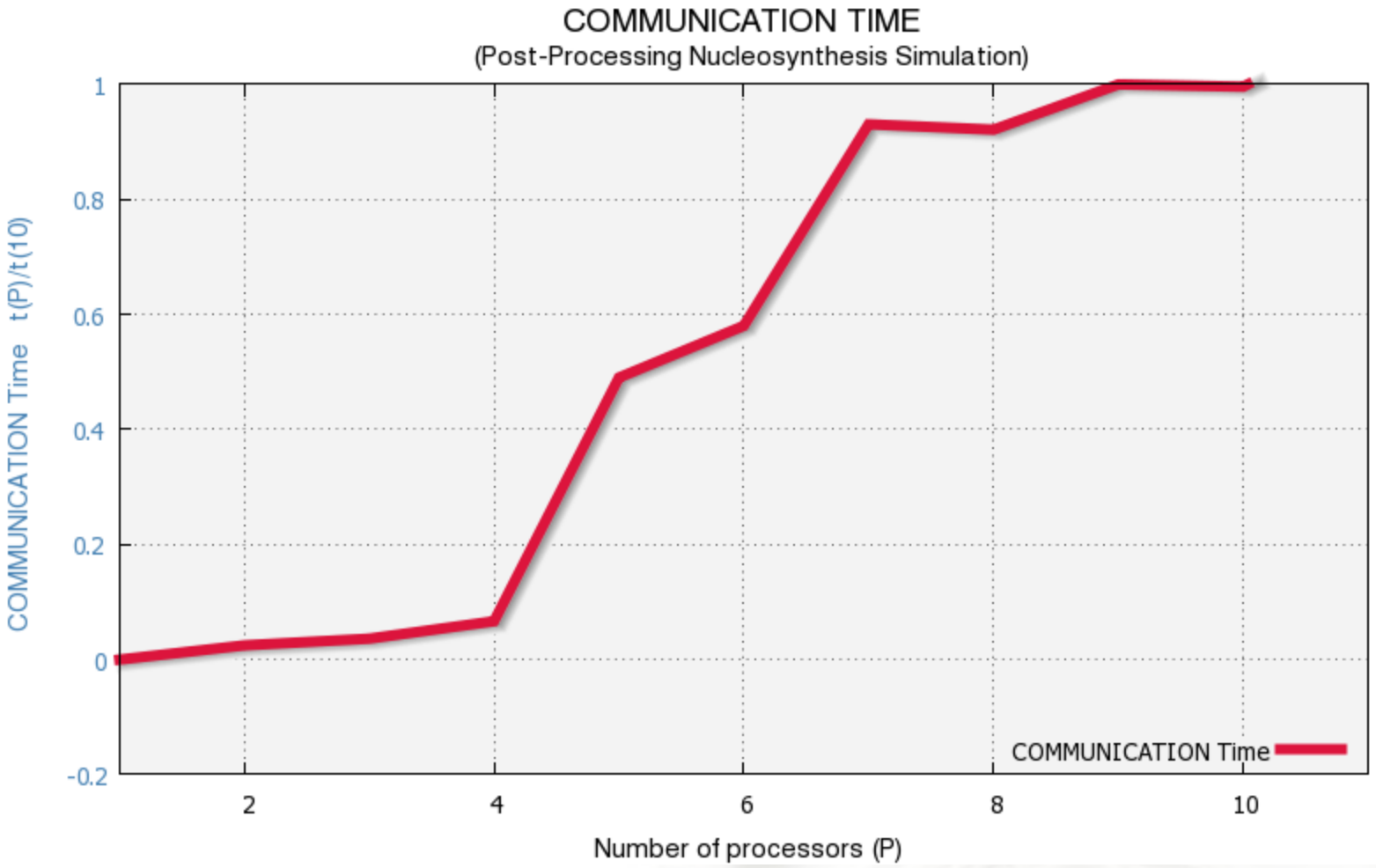




Performance: Matrix inversion time

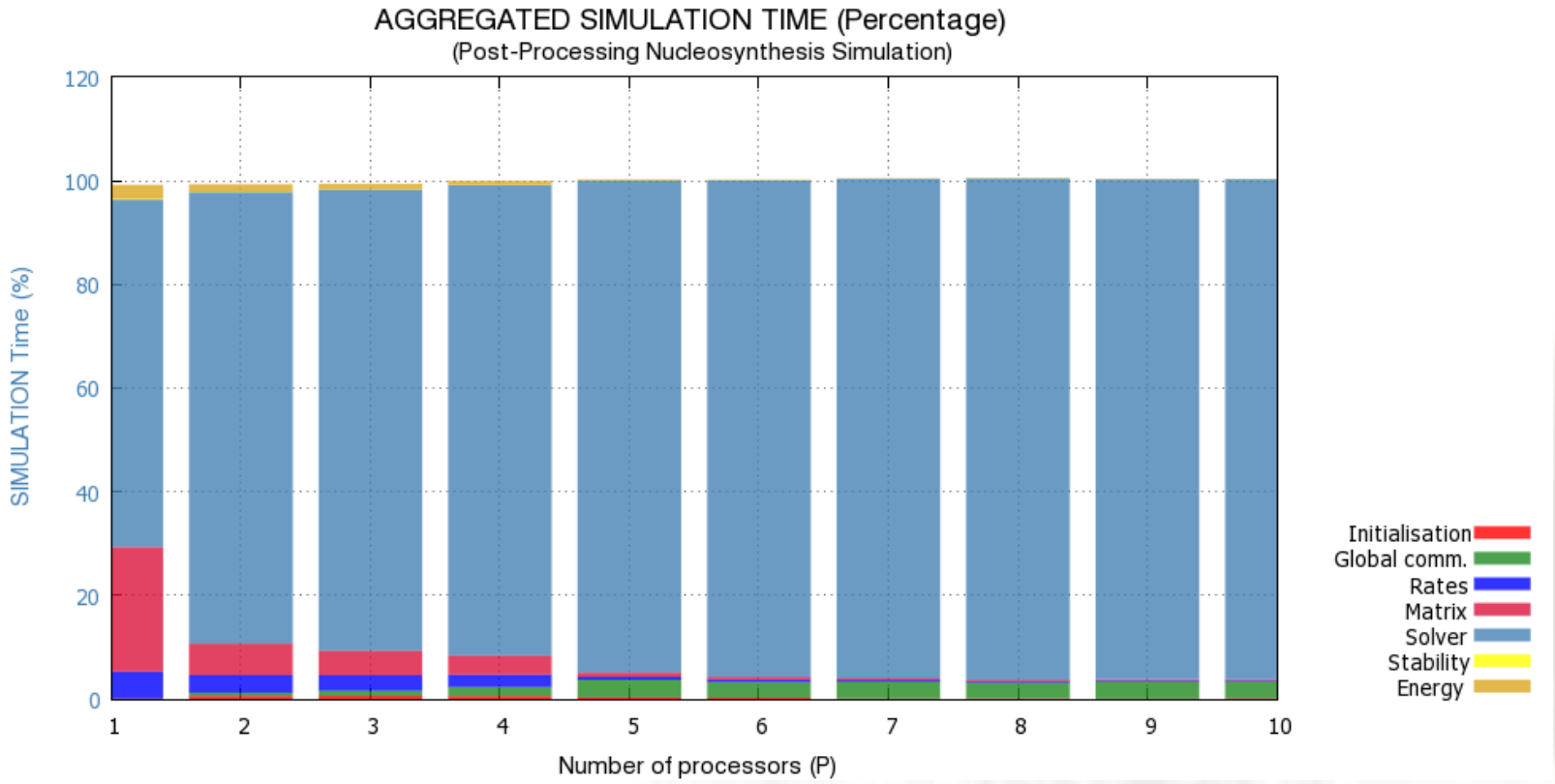


Verdict: communication cost kills performance

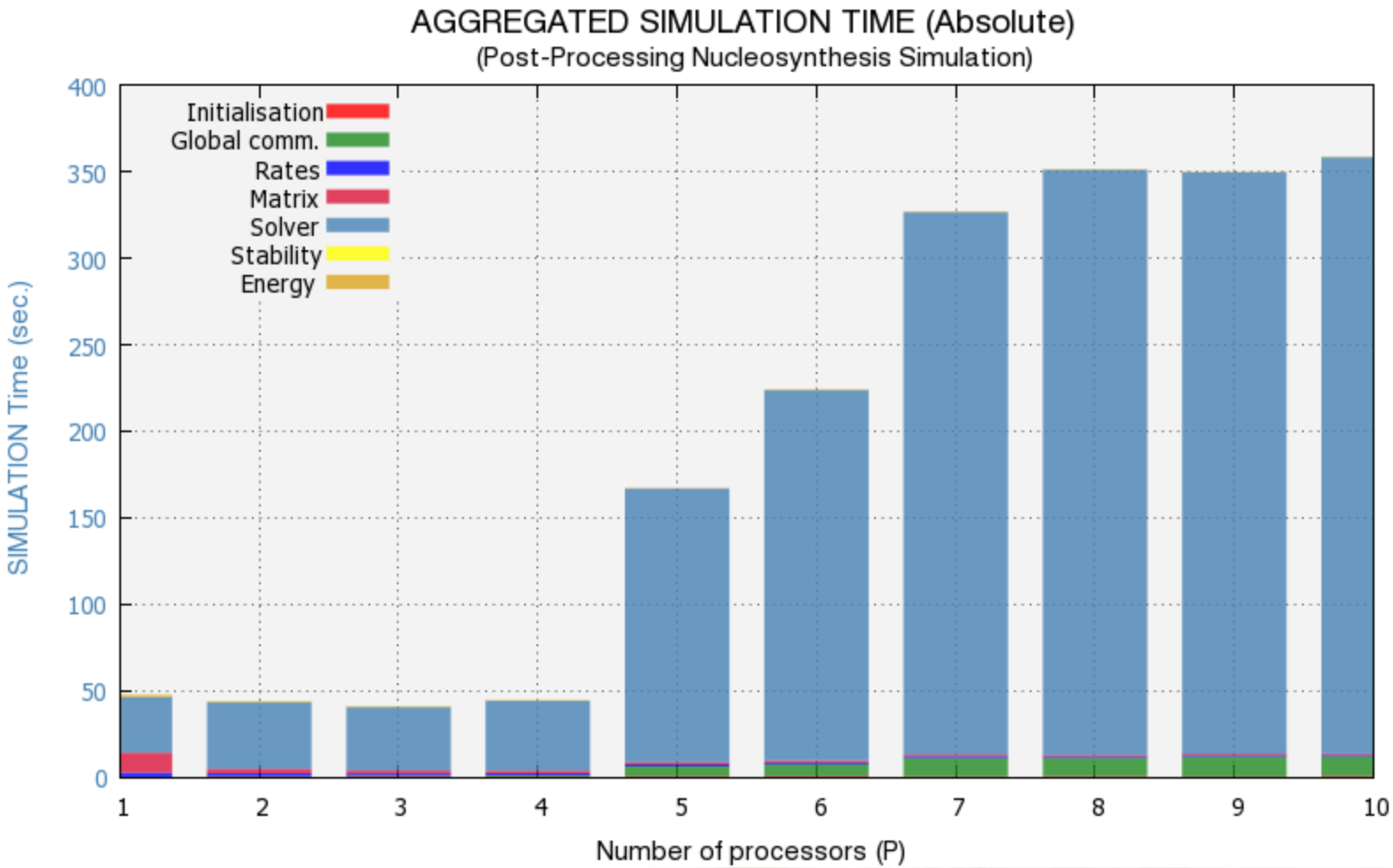


Aggregated simulation time (relative)

- The sequential execution spends most of the time inverting the matrix (82%) and building (16%) the system of equations.



Aggregated simulation time (absolute)



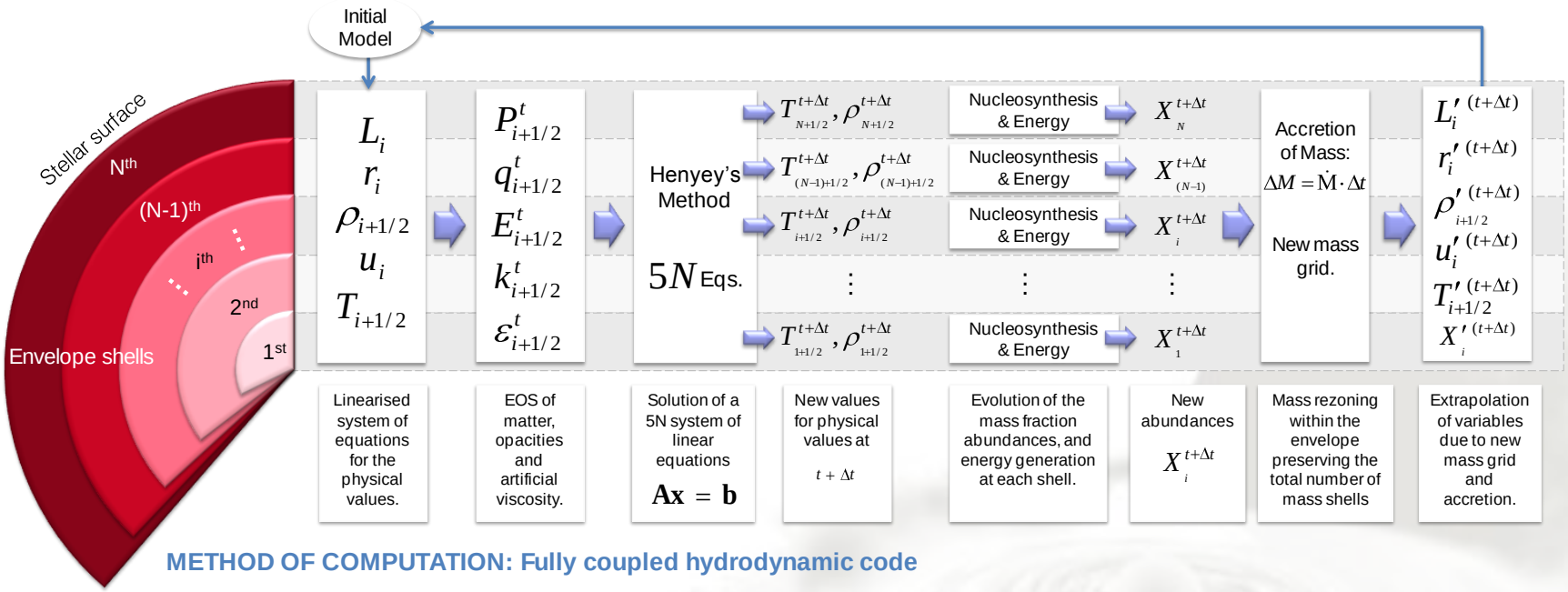
* Simulation with 5000 time-steps

Where are we?

1. Introduction
2. Designing Parallel Applications
3. Parallelisation of a Nucleosynthesis Code
4. Parallelisation of a multi-zone Hydrodynamic Code
5. Conclusions

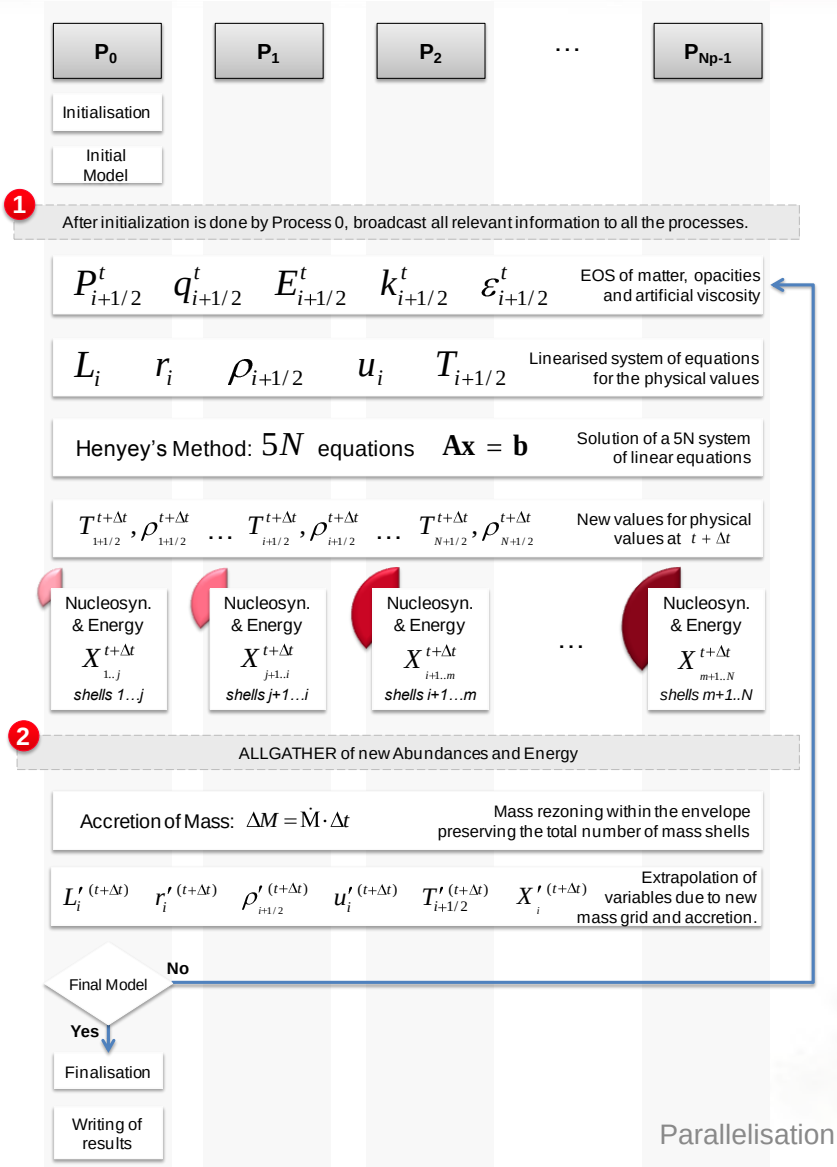
Fully coupled hydrodynamic code (SHIVA)

- In the SHIVA code, the outermost layers of the star (i.e. white dwarfs, neutron stars) are divided into **N** concentric mass shells.
- The code computes the time evolution of the luminosity **L**, the radius **r**, the velocity **v**, the temperature **T**, the density ρ, and the nuclear abundances **X** for each shell.



METHOD OF COMPUTATION: Fully coupled hydrodynamic code

SHIVA Code Parallelisation



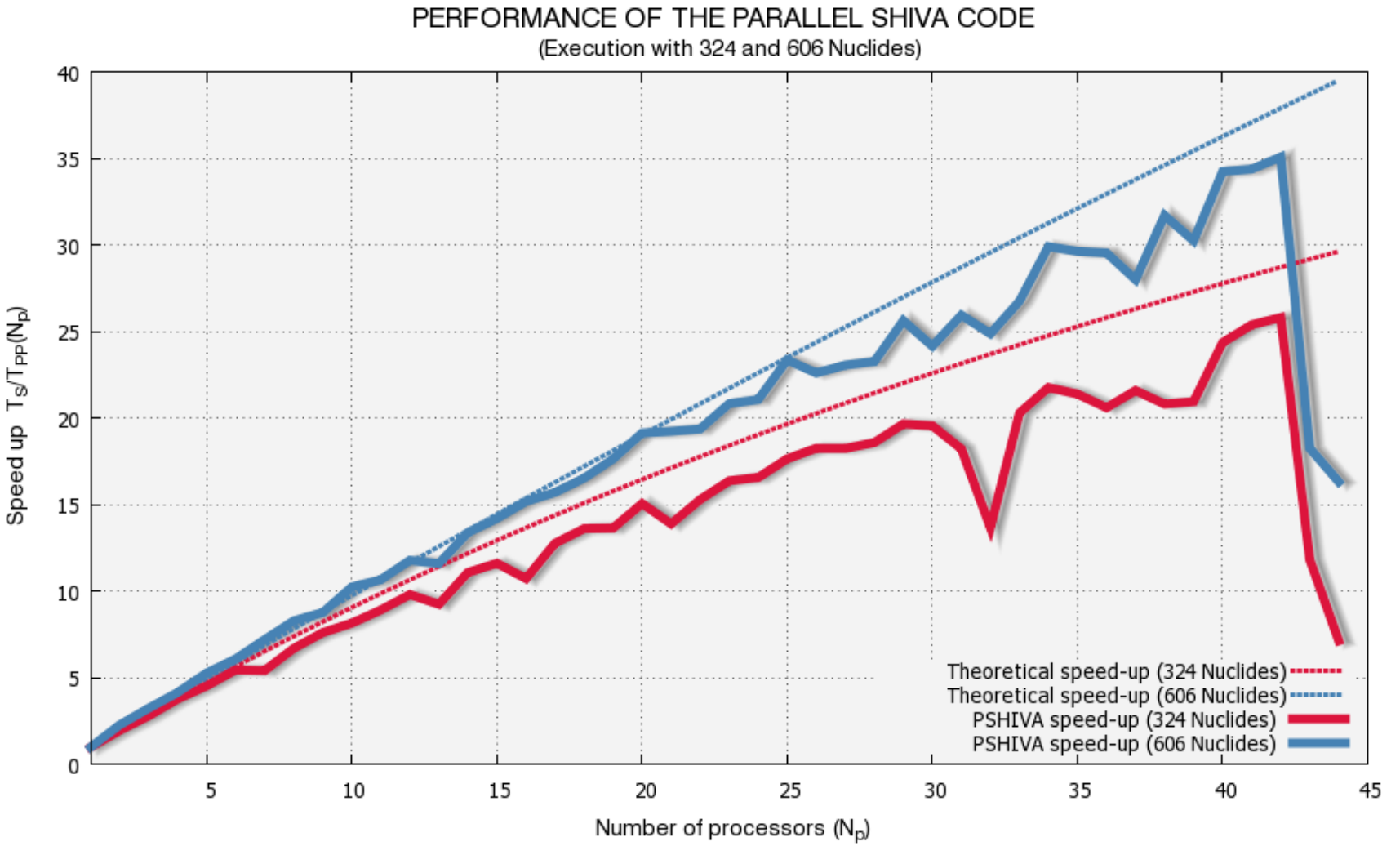
- All processors execute an instance of the hydrodynamic code, so that each process computes all processing stages (equations of matter, opacities and artificial viscosity, linearised system of equations for the physical values, solution of the system by means of the Henyey's method, etc.).
- When it comes to the computation of the nucleosynthesis and energy generated, each process performs the computation *only on a subset of all shells*.
- After this, each process broadcasts to the rest of the processors the results of the nucleosynthesis for their subset of shells.
- The execution proceeds redundantly on all nodes, all of them executing the same code with the same input data.

Performance of the parallel SHIVA code (I)

- Performance evaluation has been carried out with two different nuclear reaction networks used for Type I X-ray Bursts nucleosynthesis calculations:
  - **Reduced Simulation**: with a reduced network consisting of 324 isotopes and 1392 nuclear reactions,
  - **Extended Simulation**: with a far more complete reaction network up to 606 nuclides and 3551 nuclear reactions

	Number of shells (N)	Nuclides	Nuclear Reactions
Reduced Simulation	200	324	1392
Extended Simulation	200	606	3551

Performance of the parallel SHIVA code (II)



Performance Prediction (I)

- Each node distributes to all other processors the abundances obtained in the computation of their assigned shells.
- This represents an ALLGATHER communication procedure where all processors get the data sent by the rest of processing nodes. The communication time of this algorithm is given by (Thakur et al, 2002):

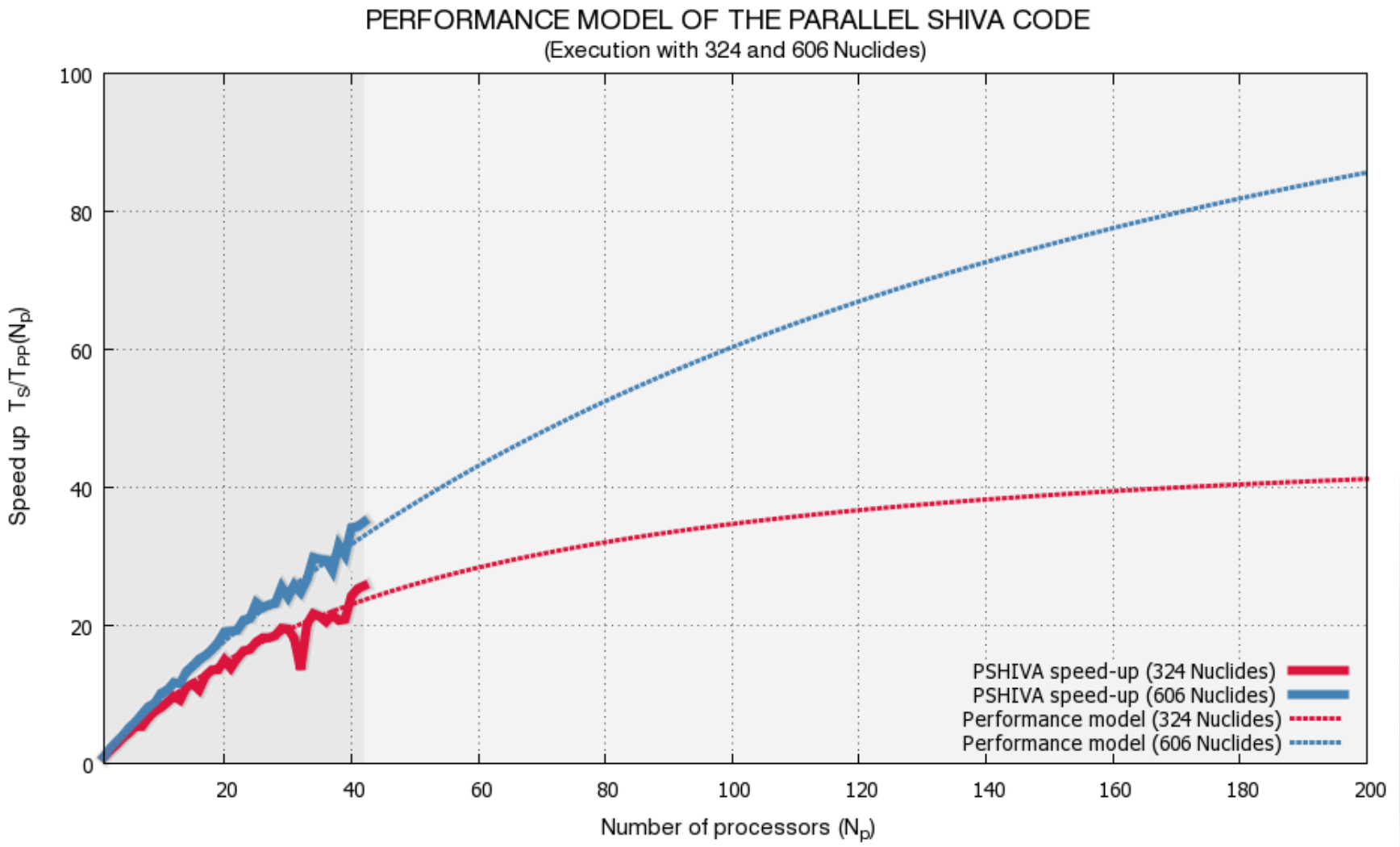
$$T_{comm} = (N_p - 1)\alpha + \frac{(N_p - 1)}{N_p} n\beta$$

n: total size of the data to be received
α: latency (or start-up time) per message
β: transfer time per byte.

- Experimental measures in the GAA's Hyperion cluster have yielded the values $\alpha = 1 \cdot 10^{-5}$ sec. and $\beta = 5 \cdot 10^{-8}$ sec.
- Incorporating T_{comm} the, the speed-up formula can be expressed as:

$$Speed - up \approx \frac{1}{(1 - p) + \left(\frac{p}{N_p}\right) + \frac{\left((N_p - 1)\alpha + \frac{(N_p - 1)}{N_p} n\beta\right)}{T_S}}$$

Performance Prediction (II)



Performance Comparison

- The parallelised version of the SHIVA code (José 1996; José & Hernanz 1998) shows excellent performance results, with significant speed-up factors accomplished in a simulation with N=200 shells.
- A speed-up factor of **26** is achieved with the reduced simulation with 42 processors. An *excellent* speed-up factor of **35** is accomplished with the extended simulation.

Code Version	Number of shells (N)	Number of time-steps	Nuclides	Nuclear Reactions	Execution time
Sequential version	200	100.000	324	1392	**~ 3 days**
			606	3551	**~ 15 days**
Parallel version (42 processors)			324	1392	**~ 3 hours**
			606	3551	**~ 10 hours**

We are almost there…

1. Introduction
2. Designing Parallel Applications
3. Parallelisation of a Nucleosynthesis Code
4. Parallelisation of a multi-zone Hydrodynamic Code
5. Conclusions

Summary and Conclusions (I)

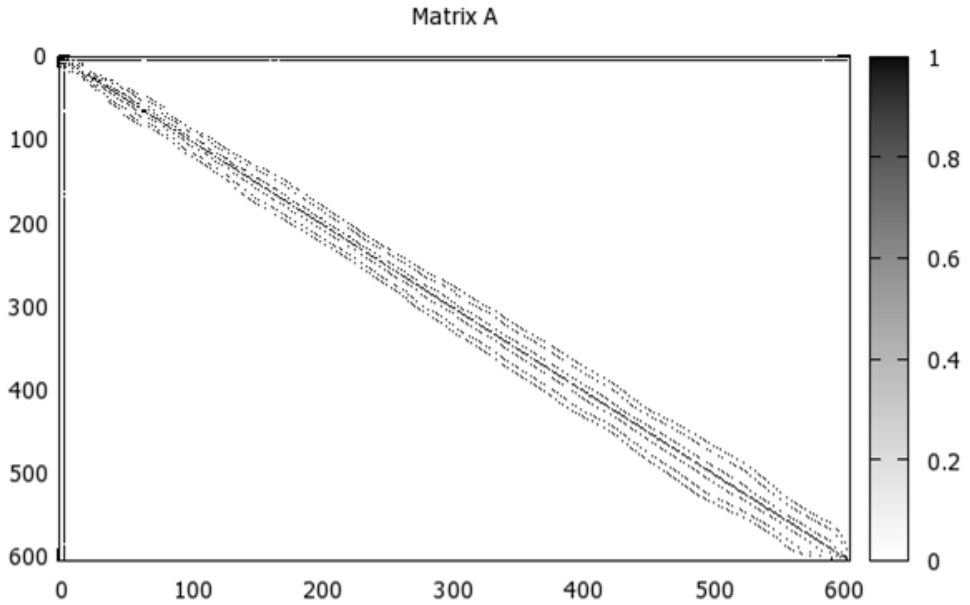
- Two numerical codes have been parallelised using the MPICH2 implementation of the Message Passing Interface (MPI) for the design of parallel applications with clusters of distributed workstations:
  - A nucleosynthesis code suitable for extensive post processing calculations, with a network containing 606 nuclides (H to ^{113}Xe) and more than 3500 nuclear reactions.
  - A one-dimensional (spherically symmetric) hydrodynamic code, in Lagrangian formulation, built originally to model classical nova outbursts (SHIVA).

Summary and Conclusions (II)

- As results have shown out, the performance of the parallel nucleosynthesis computation is much worst than the sequential, 1-node version of the code.
- This stems from the fact that the communication and message passing times between processors largely outgrow the computation time in the calculation of the parallel solution of the linearised system of equations.
- In order to increase de computation to communication time ratio (and therefore increase parallel performance), it has to be increased the order of the matrix **A**.
- It has been found that in order to get significant speed ups in the parallelisation of the nucleosynthesis calculations, the order of matrix **A** should be around 10k elements.

Order of the nucleosynthesis matrix

- The linearised system of equations of the network abundances derivatives, is a small, sparse matrix whose order is limited by the number of isotopes of the nucleosynthesis network (**NIS = 606**).



There are not so many isotopes, such that the problem size can be increased to a point where speed-ups are accomplished in the parallel solution of the system.

- It is therefore not possible to parallelise efficiently the nucleosynthesis portion of the code, and efforts in this regard should be avoided.

Summary and Conclusions (III)

- The parallelised version of the multi-zone hydrodynamic code SHIVA shows excellent performance results.
- Speed-up factors of **~26** and **~35** are achieved with the reduced and extended simulations respectively, when 42 processors are used in parallel to execute the application (with 200 shells).
- Simulations can now be executed in hours instead of days, and in days instead of months.
- Maximum speed-ups of **~40** and **~85** are predicted by the performance model when using 200 processors, for the reduced and extended simulations respectively.

Summary and Conclusions (IV)

- In preparation…

Performance Improvement of Stellar Hydro Codes. Part I: Parallelization

D. Martin[1,2⋆], J. José[1,2], and R. Longland[1,2]

[1] Departament de Física i Enginyeria Nuclear, EUETIB, Universitat Politècnica de Catalunya, c/ Comte d'Urgell 187, E-08036 Barcelona, Spain
[2] Institut d'Estudis Espacials de Catalunya (IEEC), Ed. Nexus-201, C/ Gran Capità 2-4, E-08034 Barcelona, Spain

March 8, 2013

Questions?

THANK YOU