Performance of a Parallel Global Atmospheric Chemical Tracer Model

James Demmel Sharon Smith

Computer Science Division
University of California at Berkeley

Participants

University of California, Los Angelos
C. R. Mechoso, R. P. Turco, A. Arakawa, R. Muntz, J. Spahr, J. Farrara, S. Kar

University of California, Berkeley
J. Demmel, M. Stonebraker, K. Slower, S. Smith, O. Zhakarov, B. Kralik

University of California, Santa Barbara
J. Dozier

NASA/Caltech/Jet Propulsion Laboratory
D. Halpern

Princeton University
G. Philander

Colorado State University
D. Randall

Motivation

1. Facilitate Environmental Research:
   (a) Model global atmosphere and ocean to give insight into the future of the earth's climate.
   (b) Understand and predict global warming, ozone depletion, and other phenomena such as El Niño.

2. Computational Science Research:
   (a) Performance modeling of parallel algorithms.
   (b) Developing software for complex parallel systems.

3. High Performance Computation and Communication (HPCC) Grand Challenge project funded by NASA.

Outline

1. Overview of the UCLA Earth Systems Model (ESM) and the Atmospheric Chemistry Tracer Model (GATOR)

2. Parallelizing GATOR

3. Timing model and validation

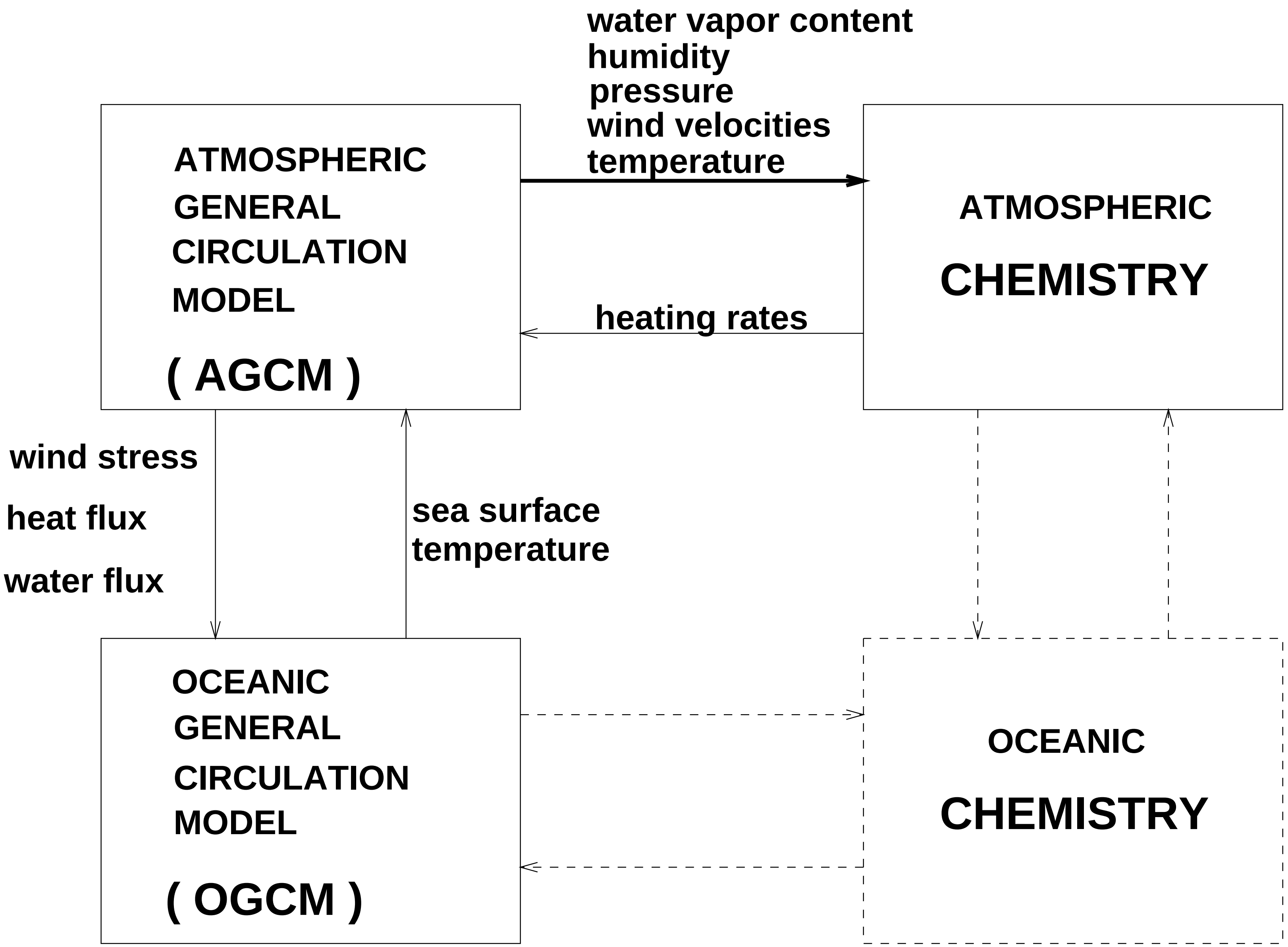
4. Timing model examples

5. Scalability of GATOR

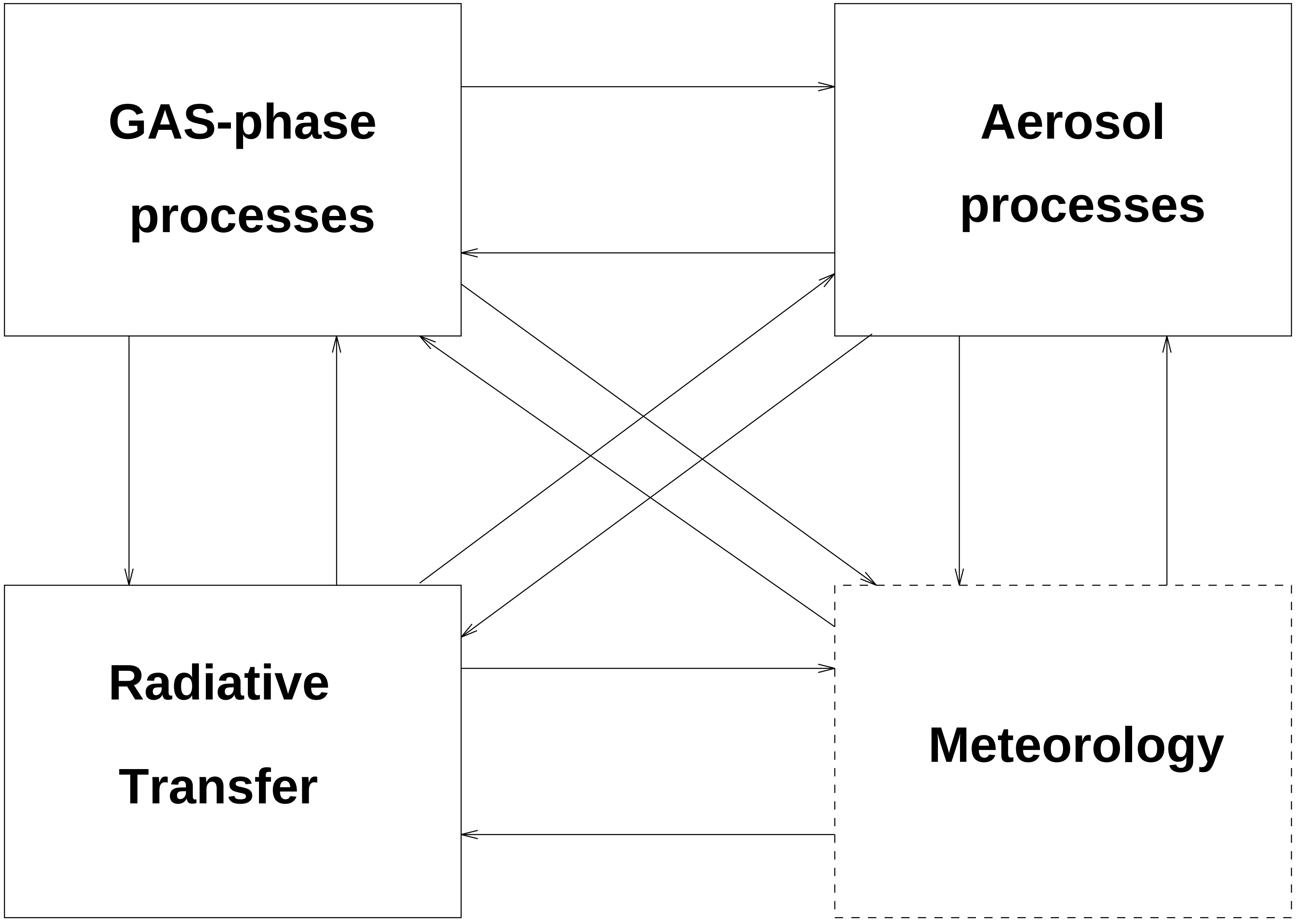
6. A look at the ESM

7. Ongoing and future work

UCLA ESM (Earth System Model) Components



Atmospheric Chemistry model "GATOR" components



Factors contributing to Computational Bottlenecks in GATOR

- Resolution of the computational grid.
- Number of chemical species.
- Amount of load imbalance in chemical kinetics equations.
- Amount of communication in transport.
- Communication of atmospheric variables from AGCM to GATOR.
- Frequency of archival of chemical species.

Data layout is important and presents tradeoff between load imbalance and high communication costs.

GATOR Algorithm (Jacobson et al.)

Focus on the two most costly parts:

- Horizontal Transport - transport chemical species between horizontal grid cells.
- Ordinary differential equation (ODE) solver - solves the chemical kinetics equations.

```
repeat
    Read input data every INPUT seconds

    Horizontal Transport
        (much communication, little computation)

    Solve ODEs
        (very parallel, modulo load balance)

    Other processes:
        Radiation transfer.
        Vertical transport.
        Deposition and rainout.
        Gas - Aerosol transfer processes.
        Aerosol processes.

    Write output data every OUTPUT seconds
until done
```

How the ODEs arise

(Reference: Jim's lecture notes and "Numerical Methods and Software" by Kahaner, Moler and Nash)

The following set of chemical reactions

$$\begin{array}{rcl} O + O2 & \overset{k1}{\rightarrow} & O3 \\ O + O3 & \overset{k2}{\rightarrow} & 2O2 \\ O2 & \overset{k3}{\rightarrow} & 2O \\ O3 & \overset{k4}{\rightarrow} & O + O2 \end{array}$$

result in the following set of ODEs, where $[O], [O2]$ and $[O3]$ are chemical concentrations:

$$\begin{aligned} d[O3]/dt &= +k_1 \cdot [O] \cdot [O2] - k_2 \cdot [O] \cdot [O3] - k_4 * [O3] \\ d[O]/dt &= -k_1 \cdot [O] \cdot [O2] - k_2 \cdot [O] \cdot [O3] \\ &\quad + 2k_3 \cdot [O2] + k_4 \cdot [O3] \\ d[O2]/dt &= -k_1 \cdot [O] \cdot [O2] + 2k_2 \cdot [O] \cdot [O3] \\ &\quad - k_3 \cdot [O2] + k_4 \cdot [O3] \end{aligned}$$

The rate constants are time varying and the solution to the ODEs changes most rapidly in the presence of sunlight.

How the ODEs are solved

Write ODEs as

$$\frac{dy}{dt} = f(t, y)$$

where

$$y = \left[\begin{array}{c} \text{[O3]} \\ \text{[O]} \\ \text{[O2]} \end{array} \right]$$

Method:

1. Use an implicit method such as the Euler backward difference method or a multistep method to calculate the values of $y_i = y(t_i) = y(i \cdot h)$ where h is the time step and $i = 1, ..., n$.

2. At each time t_i solve:

$$y_{i+1} = y_i + hf(t_{i+1}, y_{i+1})$$

for y_{i+1} using Newton's method.

Use of Netwon's Method in the ODE solver.

Solve

$$F'(y^k)(\triangle y^k) = F(y^k)$$

for $\triangle y^k$.

and then $y_{k+1} = y_k + \triangle y^k$

where

$$F(y) = y - y_i - hf(t_{i+1}, y)$$

and

$$\begin{aligned} F'(y) &= I - h\frac{\partial f(t_{i+1}, y)}{\partial y} \\ &= Jacobian \;\; of \;\; F \end{aligned}$$

Computational Bottleneck:

Solving a sparse linear system of equations with matrix F'.

Horizontal Transport

To do the full transport (both horizontal and vertical) we solve 3 1D Time split tracer continuity equations that originate from a 3D tracer continuity equation:

Given:

C - concentrations
K_i - scaled diffusion coefficients
U_x, U_y, U_z - scaled wind components
ρ^* - scaled air density

$$\begin{aligned}
\frac{\partial C}{\partial t} + \frac{\partial U_x C}{\partial X} &= \frac{\partial}{\partial X}\left(\rho^* K_1 \frac{\partial C/\rho^*}{\partial X}\right), \\
\frac{\partial C}{\partial t} + \frac{\partial U_y C}{\partial Y} &= \frac{\partial}{\partial Y}\left(\rho^* K_2 \frac{\partial C/\rho^*}{\partial Y}\right), \\
\frac{\partial C}{\partial t} + \frac{\partial U_z C}{\partial Z} &= \frac{\partial}{\partial Z}\left(\rho^* K_3 \frac{\partial C/\rho^*}{\partial Z}\right) \\
&\quad + \; vertical \;\; processes
\end{aligned}$$

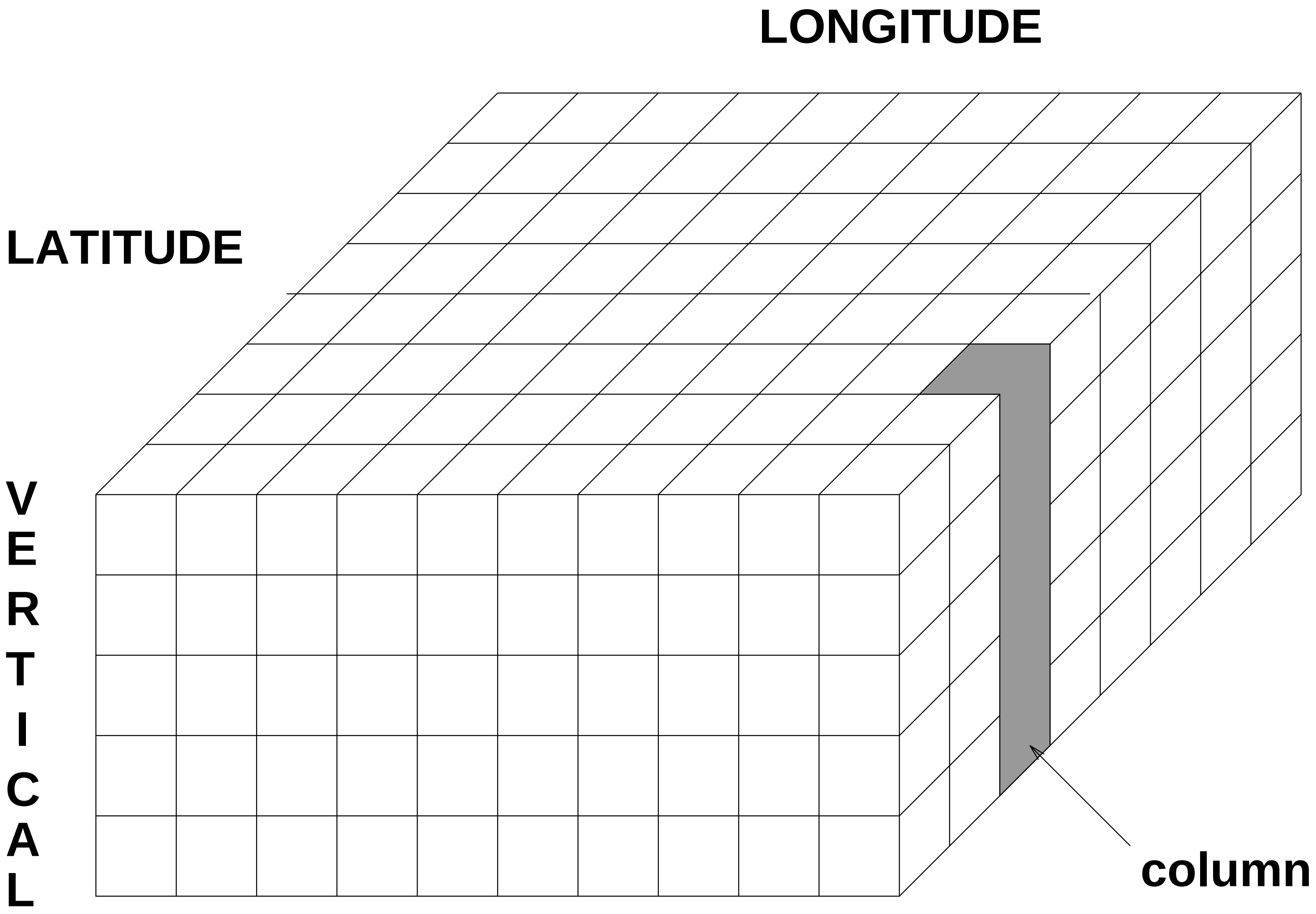
When discretized using finite elements, these model equations turn into a nearly tridiagonal system of linear equations which we solve using Gaussian Elimination.

Parallelizing GATOR:

1. Data Decomposition:
    - Assign columns of atmospheric cells to processors.
    - No further partitioning of columns because of large data movement required by vertical processes (deposition, radiation transfer, etc).
    - How should we assign columns to processors?
      Tradeoff is load balance vs. communication costs.

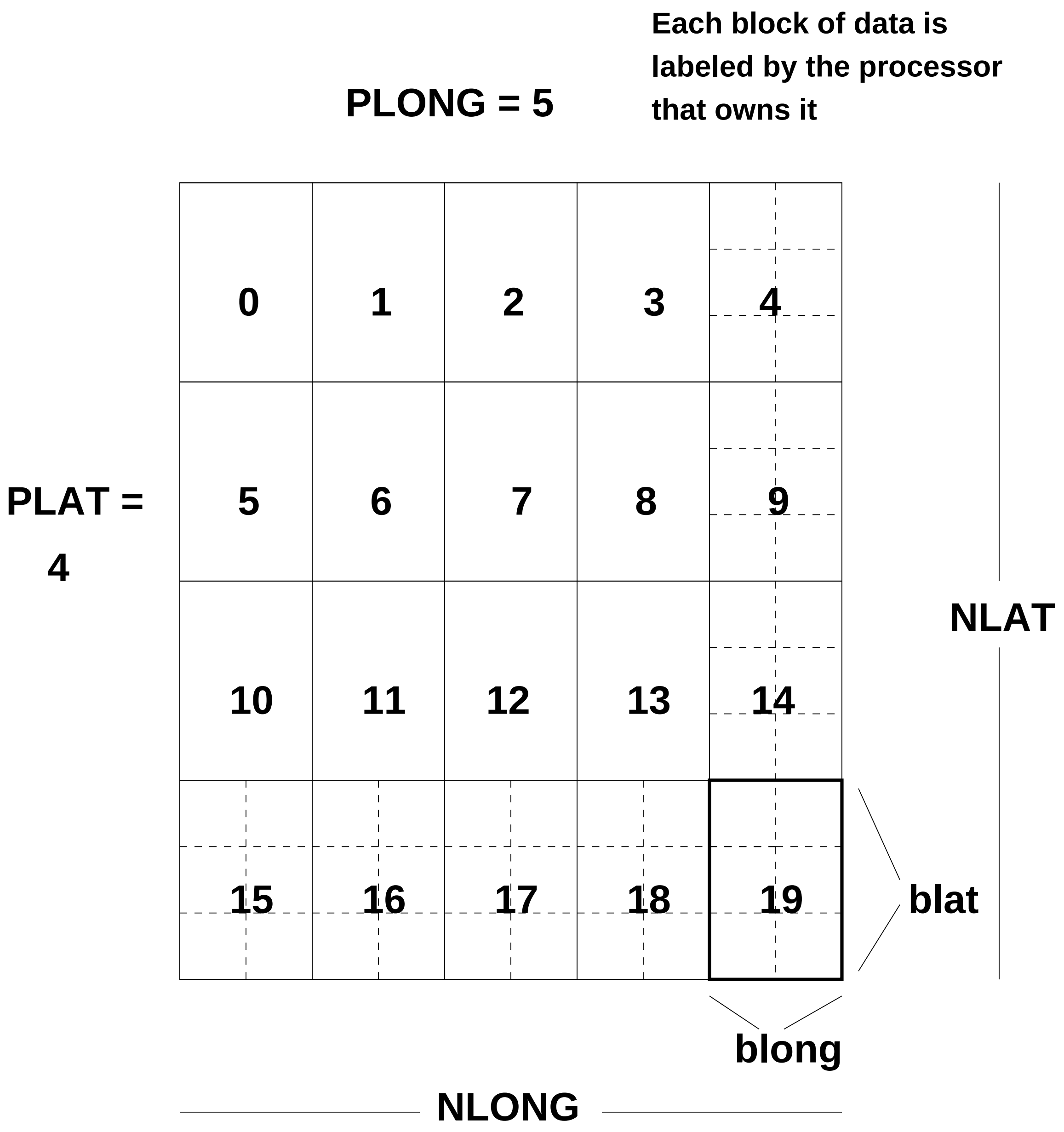
2. ODE solver:
   Each grid cell *independently* solves a sparse system of equations that arise from the chemical kinetic equations.

3. Horizontal transport:
   Use a pipeline algorithm to solve nearly tridiagonal systems of equations arising from 1D time-split tracer continuity equations.


LONGITUDE
LATITUDE
VERTICAL
column

Layout parameters for the static decomposition.

- $NLAT$, $NLONG$ - number of latitudinal and longitudinal cells.
- $PLAT$, $PLONG$ - number of processors in latitudinal and longitudinal direction
- $BLAT$, $BLONG$ - of latitudinal or longitudinal cells to a block.



BLOCK LAYOUT

Tradeoffs: Low communication costs vs. Poor load balance

PLONG = 10

PLAT = 2

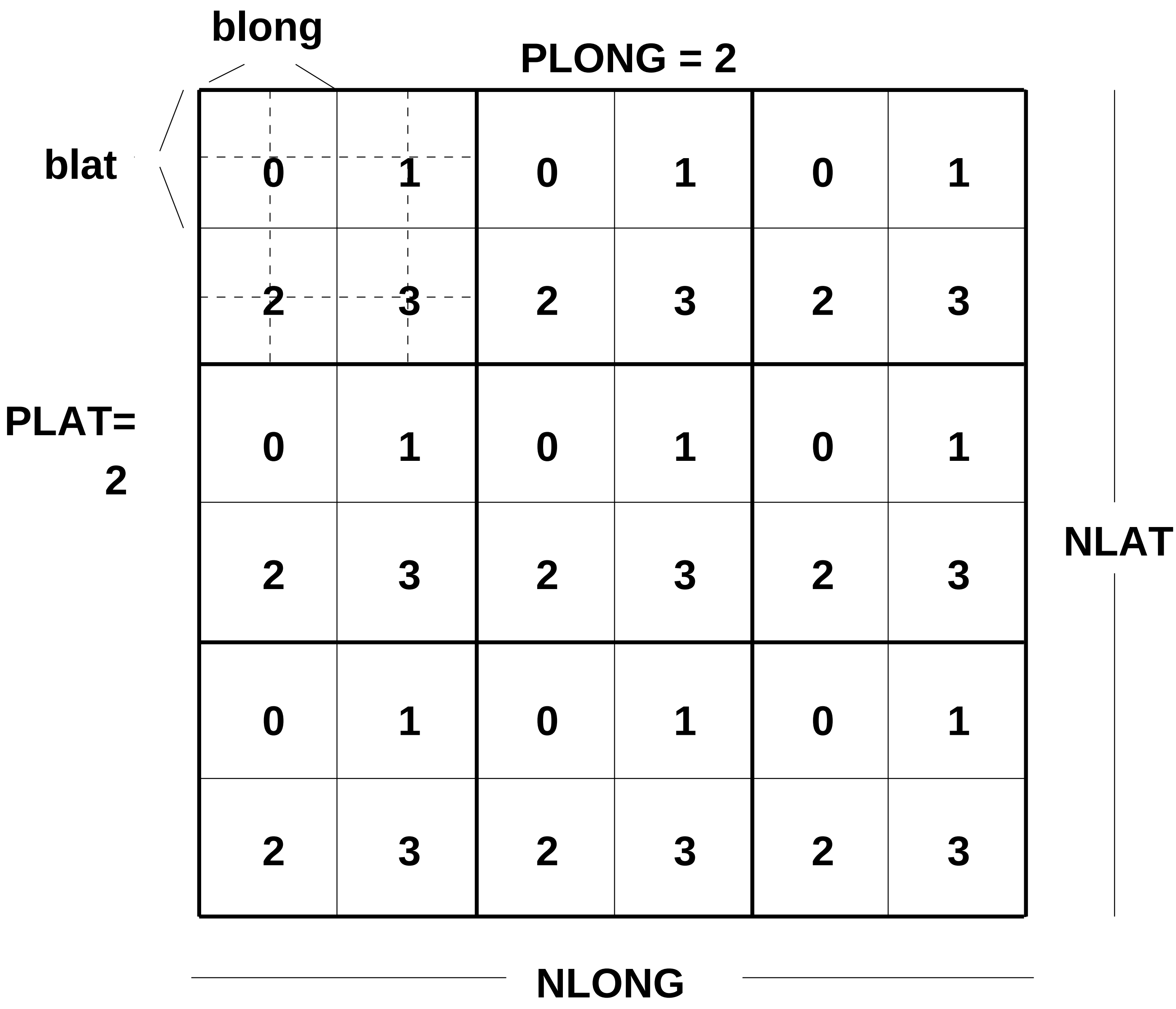
0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19

NLAT

NLONG

CYCLIC LAYOUT

Tradeoffs: High communication costs vs. Good load balance



BLOCK-CYCLIC LAYOUT

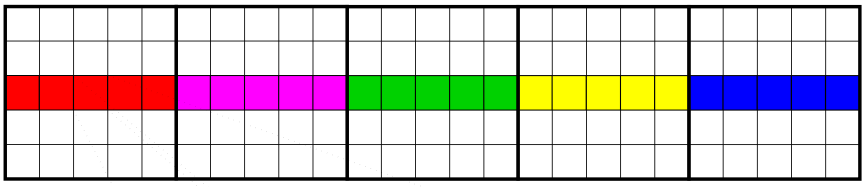
Comprimise between two extremes

Cost of solving the ODEs

	DAY	NIGHT
SUMMER	very expensive	inexpensive
WINTER	expensive	inexpensive

Horizontal Transport

How gases and aerosols are transported between grid cells and processors:

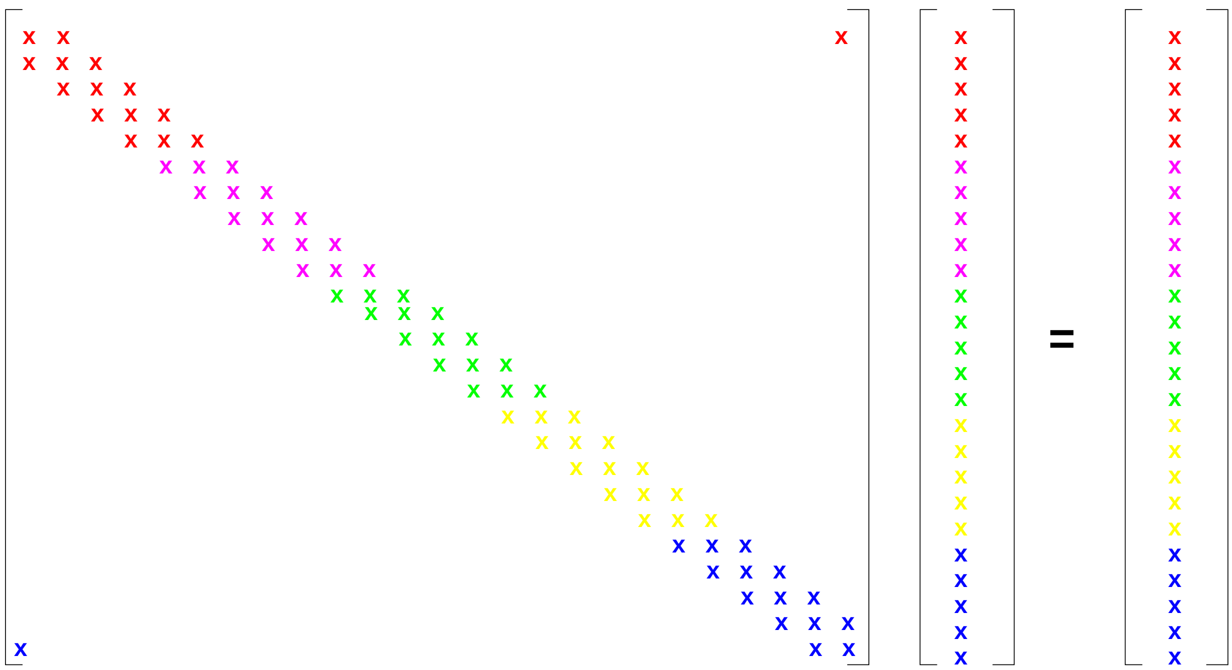


Each cell contains:

wind velocities
pressure
diffusion coeff
densities
chemical spec.

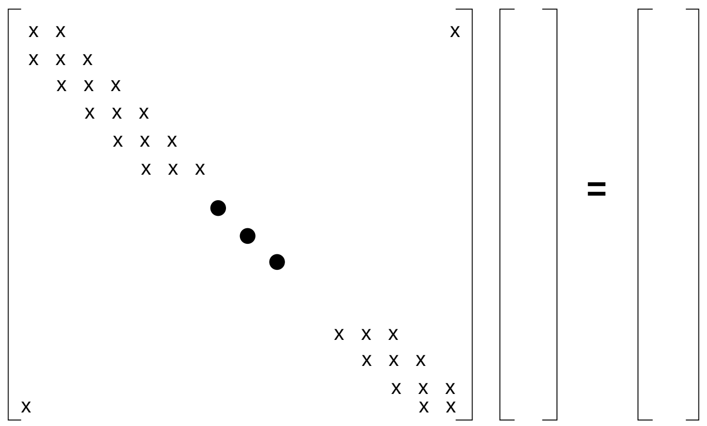
Transport from E-W involves the movement of species from one grid cell to the next adjacent grid cell.

The nearly tridiagonal system is thus formed from these elements in each of the grid cells in the direction of advection:



Horizontal Transport

Solve for new concentrations (x) using Gaussian Elimination.



A x b

For East-West Advection:

Dimension of system = NLONG = # of longitudinal cells

One system (Ax=b) for each latitude and vertical layer

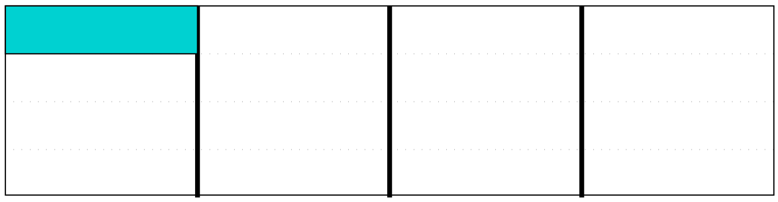
One R.H.S. b for each latitude, vertical layer and chemical species (NLAT*NVERT*NSPECIES)

Matrix A and R.H.S. b entries depend on wind, densities, and diffusion coefficients at each grid cell for a given longitudinal or latitudinal section.

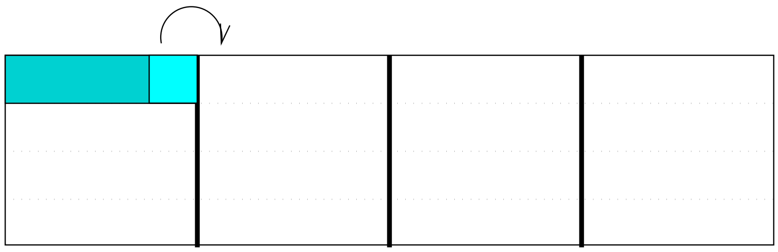
Each R.H.S. b also depends on the initial species concentrations.

Horizontal Transport Pipelining Strategy

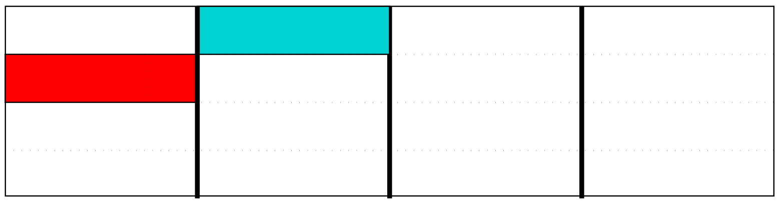
1. First processor starts the pipeline by performing the solve on some subset of its species:



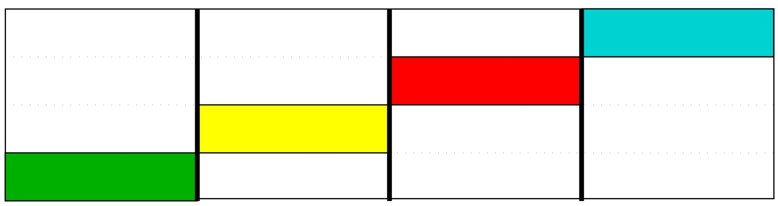
2. First processor sends the set of partial results to its neighbor.



3. First processor begins working on the next group of computations. Next processor begins work on the first set of species it receives.



4. Once the pipeline is filled, all of the processors will be busy working on some part of the solve for some subset of the species.



Modeling the Parallel Algorithm

- **Developed detailed timing models with application, layout, and machine parameters.**
- **Description of model parameters and abbreviations**
  - $NLAT$, $NLONG$ - # of lat and long cells
  - $NVS = NVERT \times NSPECIES$
  - $PLAT$, $PLONG$ (# of processors in lat and long direction)
  - $BLAT$, $BLONG$ # of cells to a block
  - $NBYTES$ - number of bytes per word
  - lat - comm. latency and overhead period
  - bw - comm. bandwidth (Mbytes/sec)
  - m - Mflop rate of processor (Mflops/sec)
  - $trans$ - Number of floating point operations in transport inner loop
  - ode - Number of floating point operations per grid cell in ODE solver

Cost of ODE Solver

$$
\begin{aligned}
T_{ode} &= \frac{Total\ \ Flops}{\#\ \ Processors \times flop\ \ rate} \\
&= \frac{NLAT \times NLONG \times NVS \times ode}{PLAT \times PLONG \times m}
\end{aligned}
$$

Total Cost of Horizontal Transport

$$
T_{trans} = T^{EW}_{initial} + T^{EW}_{solve} + T^{NS}_{initial} + T^{NS}_{solve}
$$

Initialization Cost in EW Horizontal Transport

(setting up the R.H.S for the solve)

- **Number of messages:**

$$i\_xchange \cdot NS$$

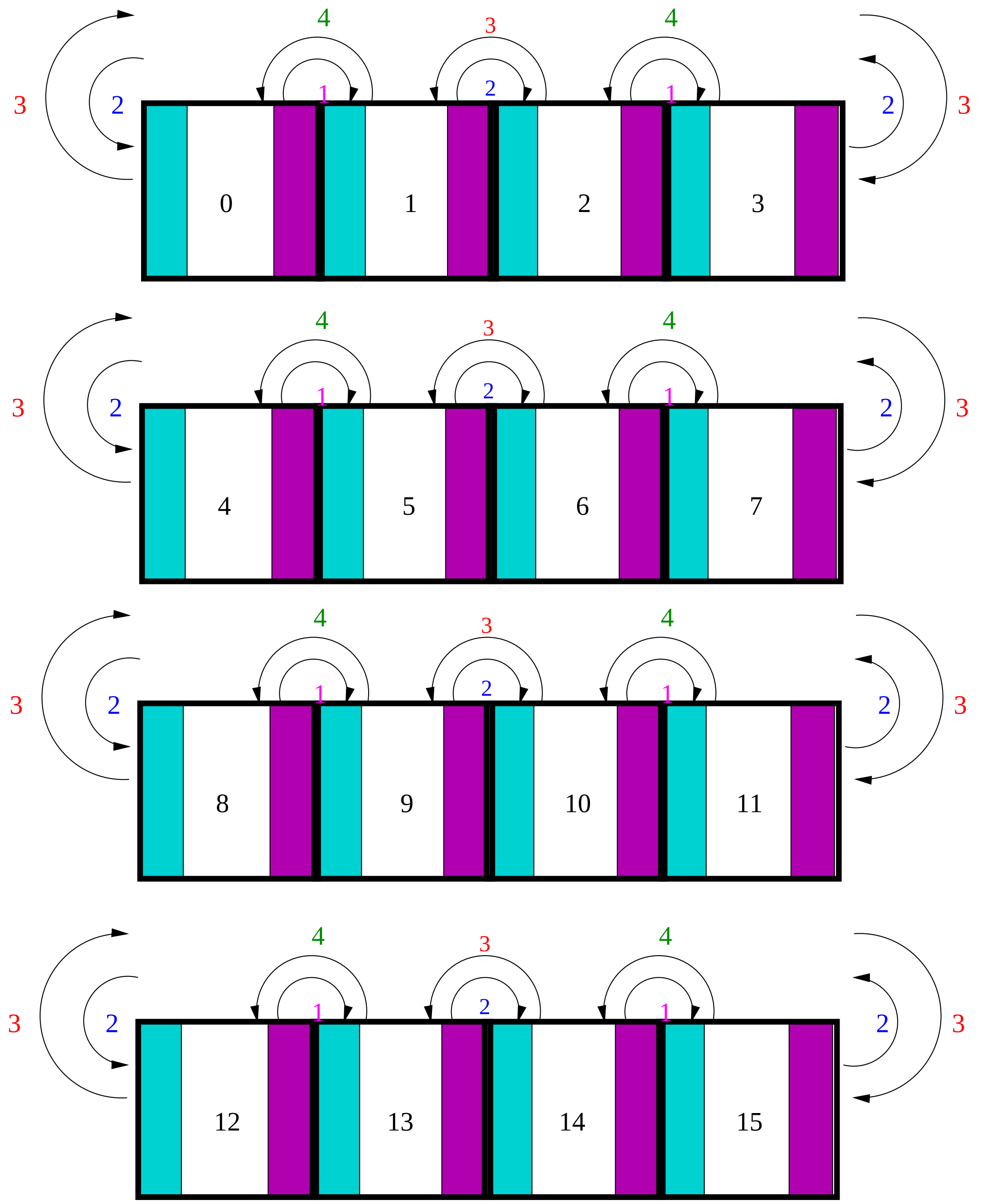
- **Size of each message:**

$$\frac{\left\lceil \frac{NLAT}{PLAT} \right\rceil \cdot NVS \cdot NBYTES}{NS}$$

- **Total time:**

$$i\_xchange \cdot NS \cdot lat$$
$$+ \quad \frac{i\_xchange \cdot \left\lceil \frac{NLAT}{PLAT} \right\rceil \cdot NVS \cdot NBYTES}{bw}$$

Order of Initialization Communication in EW Horizontal Transport



Cost of Nearly Tridiagonal Solve in EW Horizontal Transport (Global)

- **Number of messages:**

$$2N\_mesg \cdot p\_xchange$$

- **Size of each message:**

$$\frac{\left\lceil \frac{2 \cdot NLAT}{PLAT} \right\rceil \cdot NVS \cdot NBYTES}{nr}$$

- **Number of flops between messages:**

$$\frac{BLONG \left\lceil \frac{NLAT}{PLAT} \right\rceil NVS \cdot trans}{nr}$$

- **Total time:**

$$\begin{aligned}
& 2N_mesg \cdot p_xchange \cdot lat \\
+ \quad & 2N_mesg \cdot p_xchange \frac{\left\lceil \frac{2 \cdot NLAT}{PLAT} \right\rceil \cdot NVS \cdot NBYTES}{nr \cdot bw} \\
+ \quad & 2N_mesg \cdot p_xchange \frac{BLONG \left\lceil \frac{2 \cdot NLAT}{PLAT} \right\rceil NVS}{nr} \frac{trans}{m}
\end{aligned}$$

Measured Performance vs. Timing Model Estimates for the LA Basin

- **UCB CM-5 using CMMD message passing and no vector units.**
  ($lat = 100\mu$ seconds; $bw = 5MB/s$)

- **UCB SP-1 with PVM.**
  ($lat = 3000\mu$ seconds; $bw = 1-2MB/s$)

- **MAUI SP-2 with PVM on the switch.**
  ($lat = 1000\mu$ seconds; $bw = 5-6MB/s$)

- **NAS SP-2 with PVMe.**
  ($lat = 450\mu$ seconds; $bw = 31MB/s$)

Architecture	Hortran		ODE Solver	
	Meas.	Est.	Meas.	Est.
UCB CM-5 (16 procs)	123 s	138 s	*5783 s*	5760 s
UCB SP-1 (8 procs)	*528 s*	637 s	264 s	230 s
MAUI SP-2 (8 procs)	117 s	169 s	*181 s*	153 s
MAUI SP-2 (16 procs)	*145 s*	207 s	94 s	77 s
NAS SP-2 (8 procs)	42 s	69 s	*102 s*	92 s
NAS SP-2 (16 procs)	49 s	88 s	53 s	46 s

Computational bottlenecks are italicized (except where the computations are almost balanced).

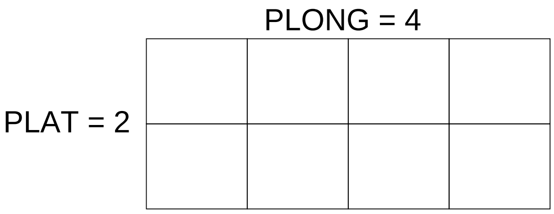
Timing Model Debugging Example:

How increasing P can increase runtime in the horizontal transport.

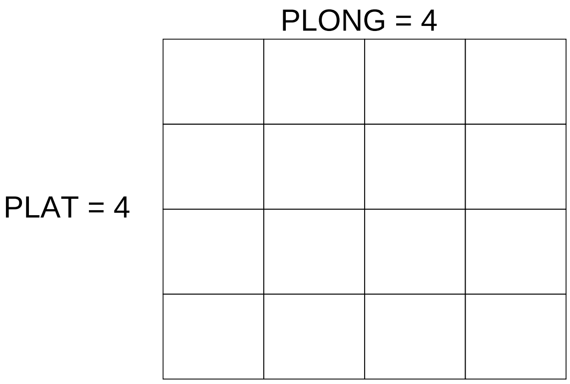
- **Originally had a constant and not a variable for** *p_xchange* **in the number of messages for the nearly tridiagonal solve:**

$$2\, N\_mesg \cdot p\_xchange$$

- **For an 8 processor (2x4) layout,** $p\_xchange = 1$ **in the N-S horizontal transport:**



- **For a 16 processor (4x4) layout,** $p\_xchange = 2$ **in the N-S horizontal transport:**



- **The measure cost for the 16 processor transport was larger than for the 8 processor layout**

- **Timing model didn't initially reflect this anomaly.**

Timing model debugging and performance tuning: Changing number of messages (NS and NR) to tune performance

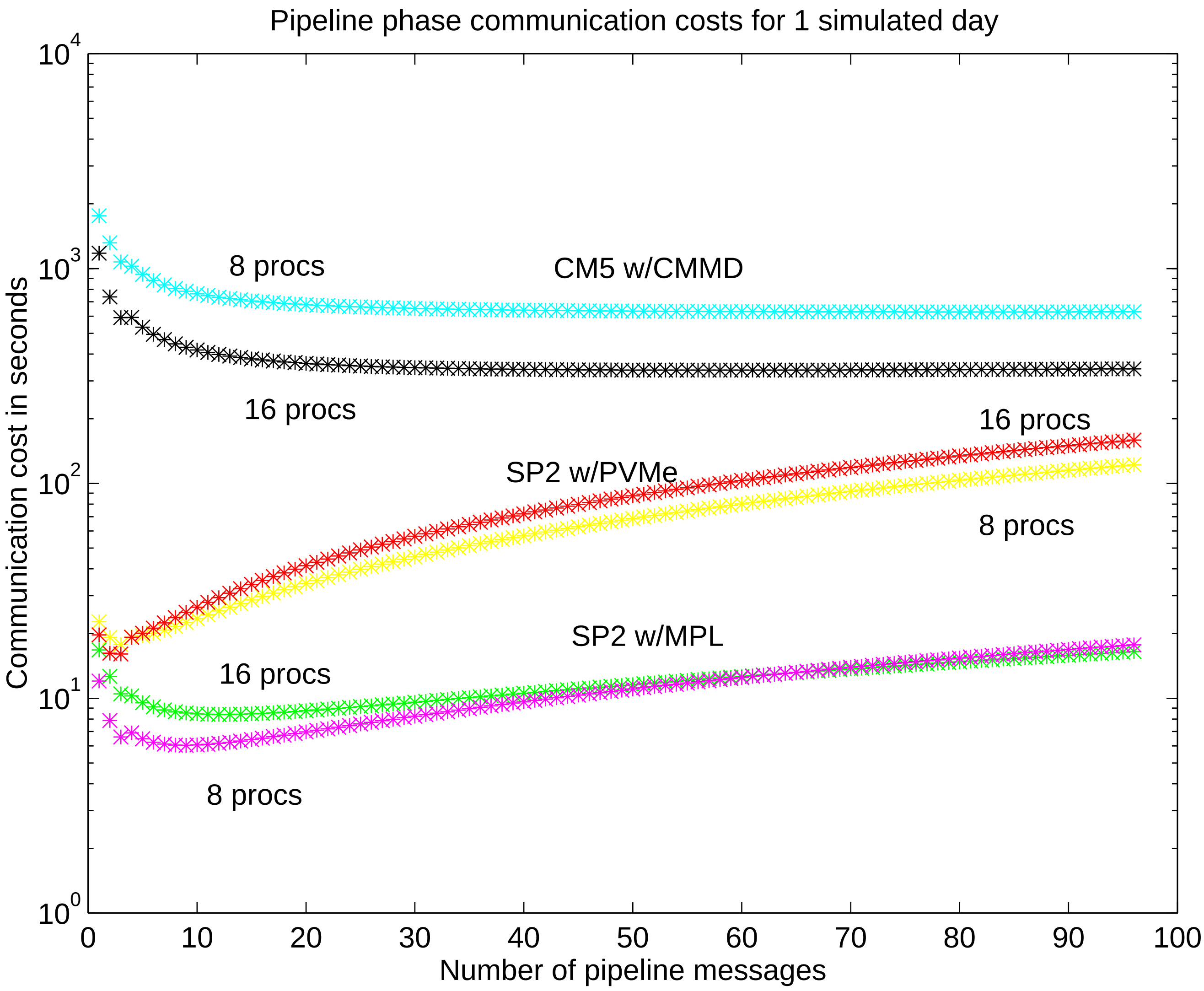
- Originally the timing model did not model the cost of setting up the R.H.S. of the equation for the nearly-tridiagonal solve, and this was significant for PVM.

- Improvement in performance by more than a factor of 4 with combined tuning of NR and NS for the pipelining and R.H.S. setup.

# Procs	CASE	HORTRAN
8	NR = 96 NS = 96	65 s
	NR = 8 NS = 96	40 s
	NR = 8 NS = 1	15 s
16	NR = 96 NS = 96	74 s
	NR = 8 NS = 96	47 s
	NR = 8 NS = 1	17 s

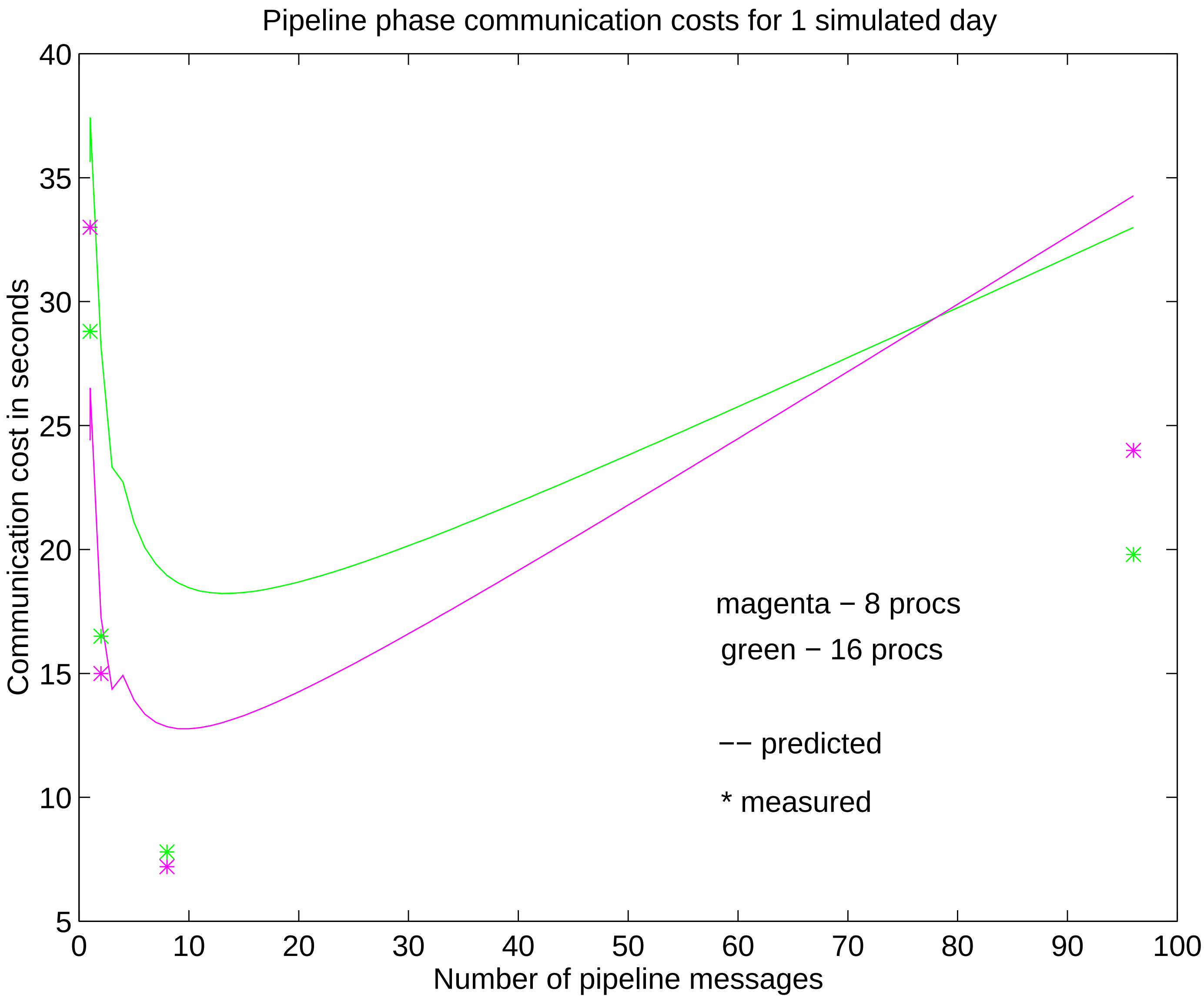
Performance Tuning Example

Can use the timing model to determine the optimal number of messages to use in the pipelined solve phase of the Horizontal Transport.



Performance Tuning Example
(Estimated vs. Measured times)

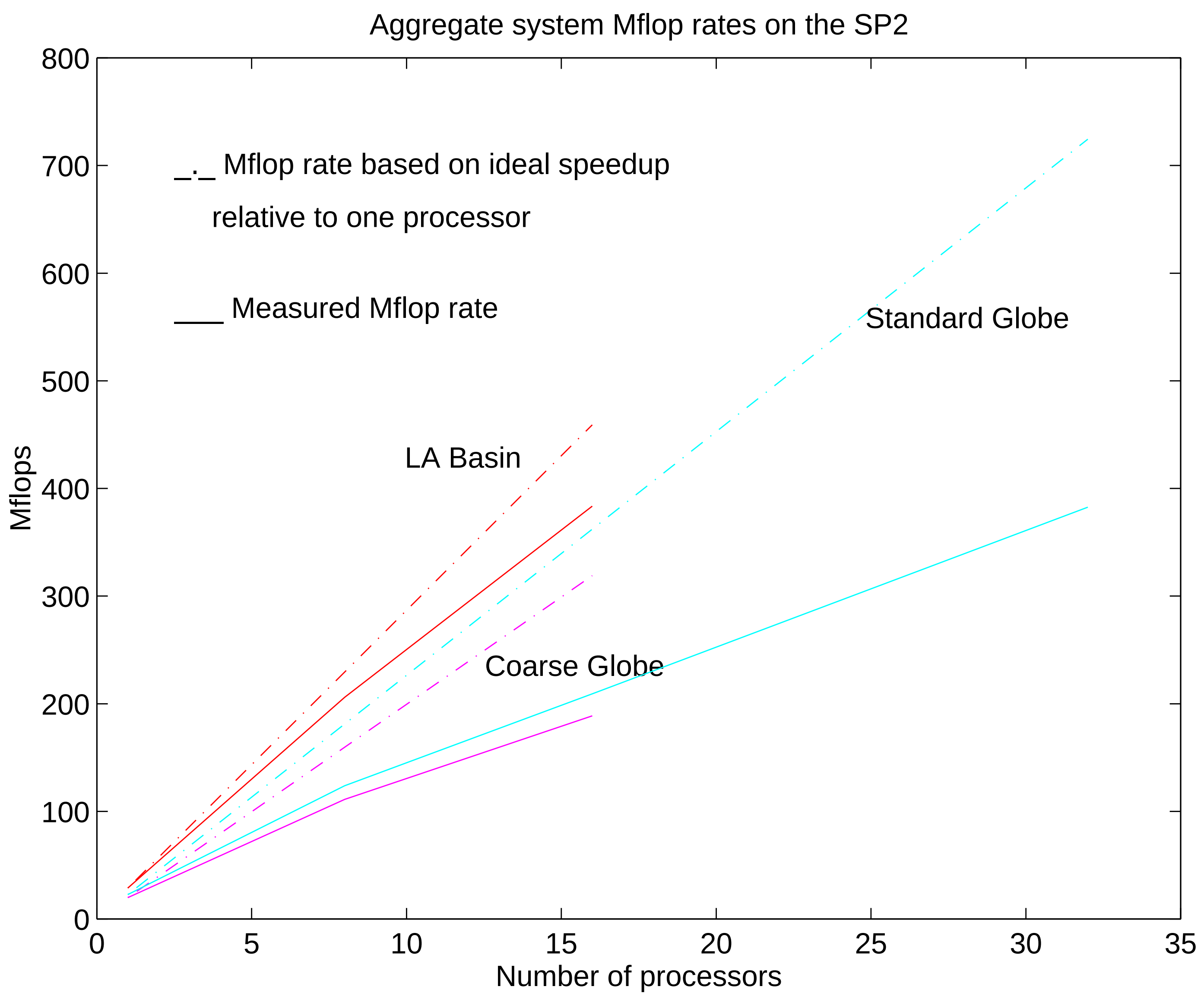
- MAUI SP2 with MPL



Scalability of GATOR

- **LA Basin**
  - $20\ lat \times 20\ long \times 5 vert$;
  - **96 chemical species;**
  - **ODEs solved every 30 simulated minutes; and**
  - **transport done every 5 simulated minutes**
- **Coarse resolution globe:**
  - $20\ lat \times 20\ long \times 5 vert$;
  - **96 chemical species;**
  - **ODEs solved every 15 simulated minutes; and**
  - **transport done every 5 simulated minutes**
- **Standard resolution globe:**
  - $44\ lat \times 72\ long \times 5 vert$;
  - **96 chemical species;**
  - **ODEs solved every 15 simulated minutes; and**
  - **transport done every 5 simulated minutes**

Scalability of GATOR



Speedups

Prob.	Arch.	HOR.	ODE	SP(C90)
	SP2-8 PVMe	3.4	7.4	1.0
LA	SP2-16 PVMe	4.2	14	1.8
	SP2-8 MPL	3.8	7.4	1.1
LA	SP2-16 MPL	5.3	14	1.9
coarse	SP2-8 PVMe	2.9	5.7	1.1
GLOB	SP2-16 PVMe	2.2	10.9	1.8
	SP2-8 PVMe	4.8	5.0	1.0
fine	SP2-16 PVMe	2.4	8.8	1.5
GLOB	SP2-32 PVMe	3.7	17.5	2.9

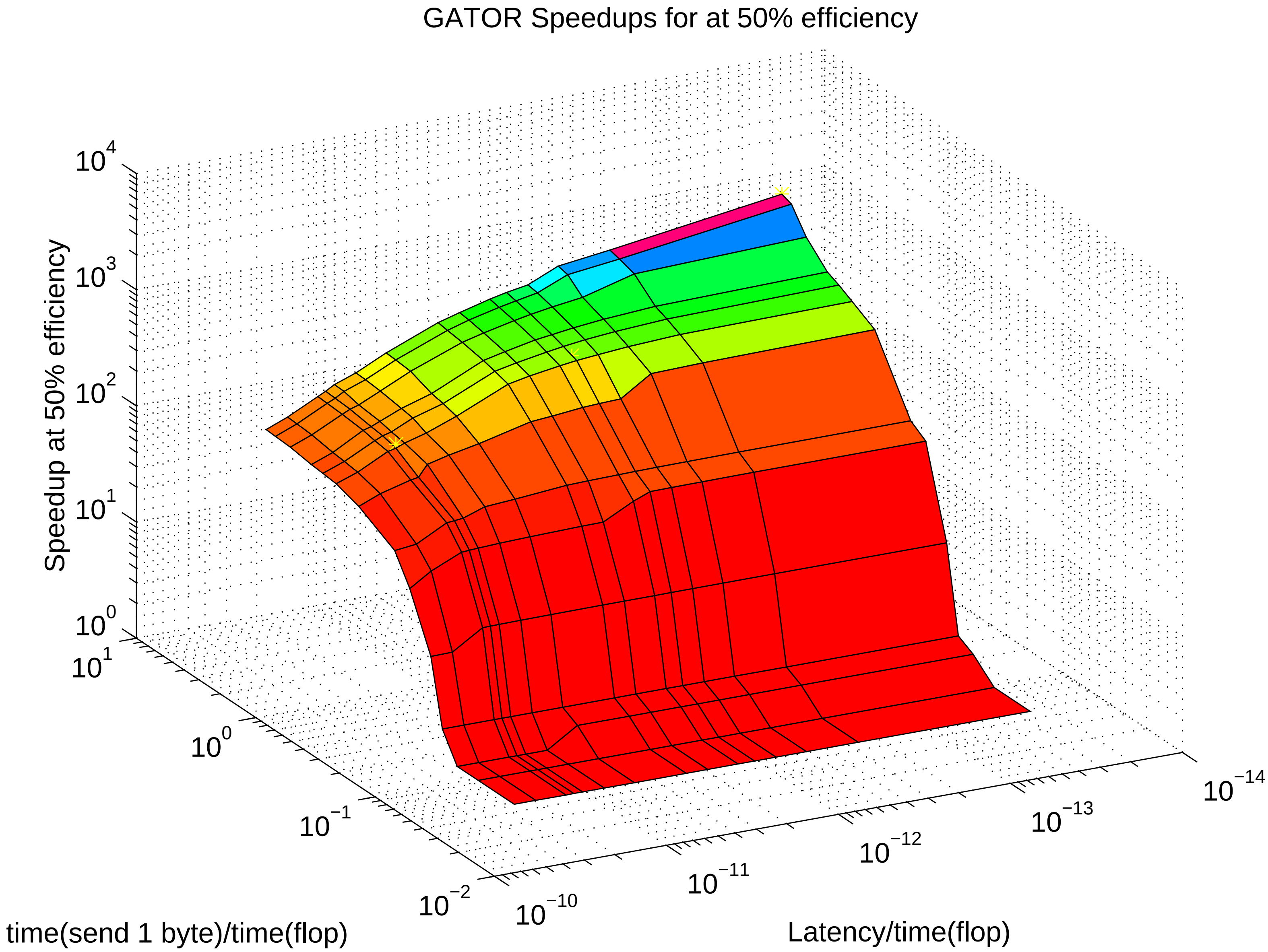
- SP(C90) is the speedup of the transport and the ODE solver on the SP2 relative to a 1 processor C90.

- SP2 results using fast memory ('fat nodes').

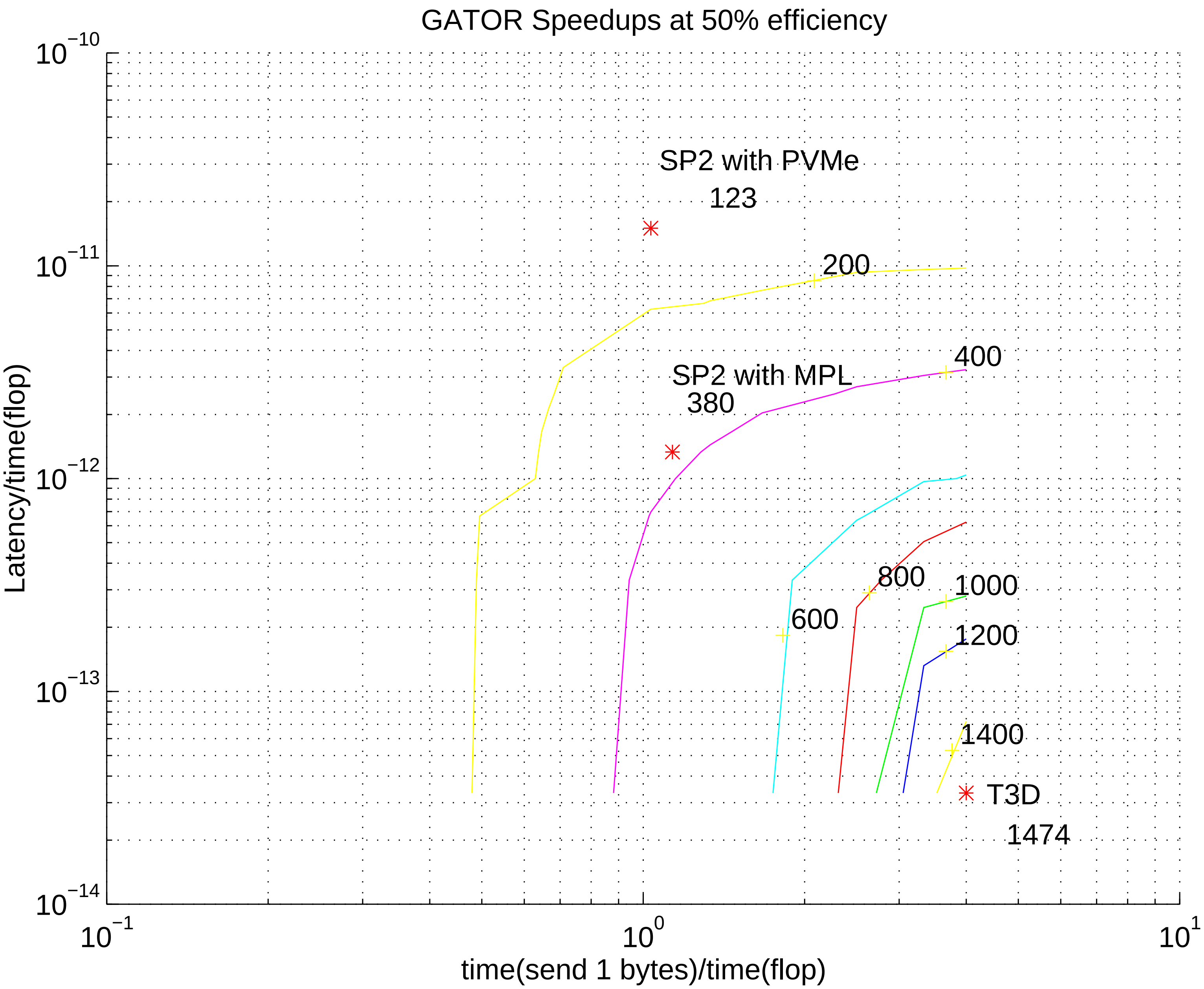
- LA Basin scalability limited by communication costs.

- Global scalability limited by both communication and load imbalance.

Sensitivity of GATOR speedups to latency and bandwidth



Scalability of GATOR (cont.)
(Speedup at 50 % Efficiency)



Load imbalance of the ODE solver

Block mapping: ratio of the most to least heavily loaded processor is 3.6

Improving the load balance of the ODE solver

Block-cyclic mapping: load imbalance ratio of the most to least heavily loaded processor is 2

Block mapping: load imbalance ratio = 3.6

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

32 Processor original block layout

0	1	2	3	0	1	2	3
4	5	6	7	4	5	6	7
0	1	2	3	0	1	2	3
4	5	6	7	4	5	6	7

8 processor block-cyclic layout

0	1	2	3
4	5	6	7

8 processor block-layout

Conclusions concerning GATOR stand-alone performance

- GATOR parallel performance for LA Basin and Globe equals or beats that of a single processor C90 for 8 processors or more.
- Timing model has been validated against parallel implementation and exhibits reasonable agreement with measured times.
- Use of the timing model in performance tuning led to a factor of 4 reduction in communication costs.
- Block-cyclic layout should decrease load imbalance by a factor of 2.