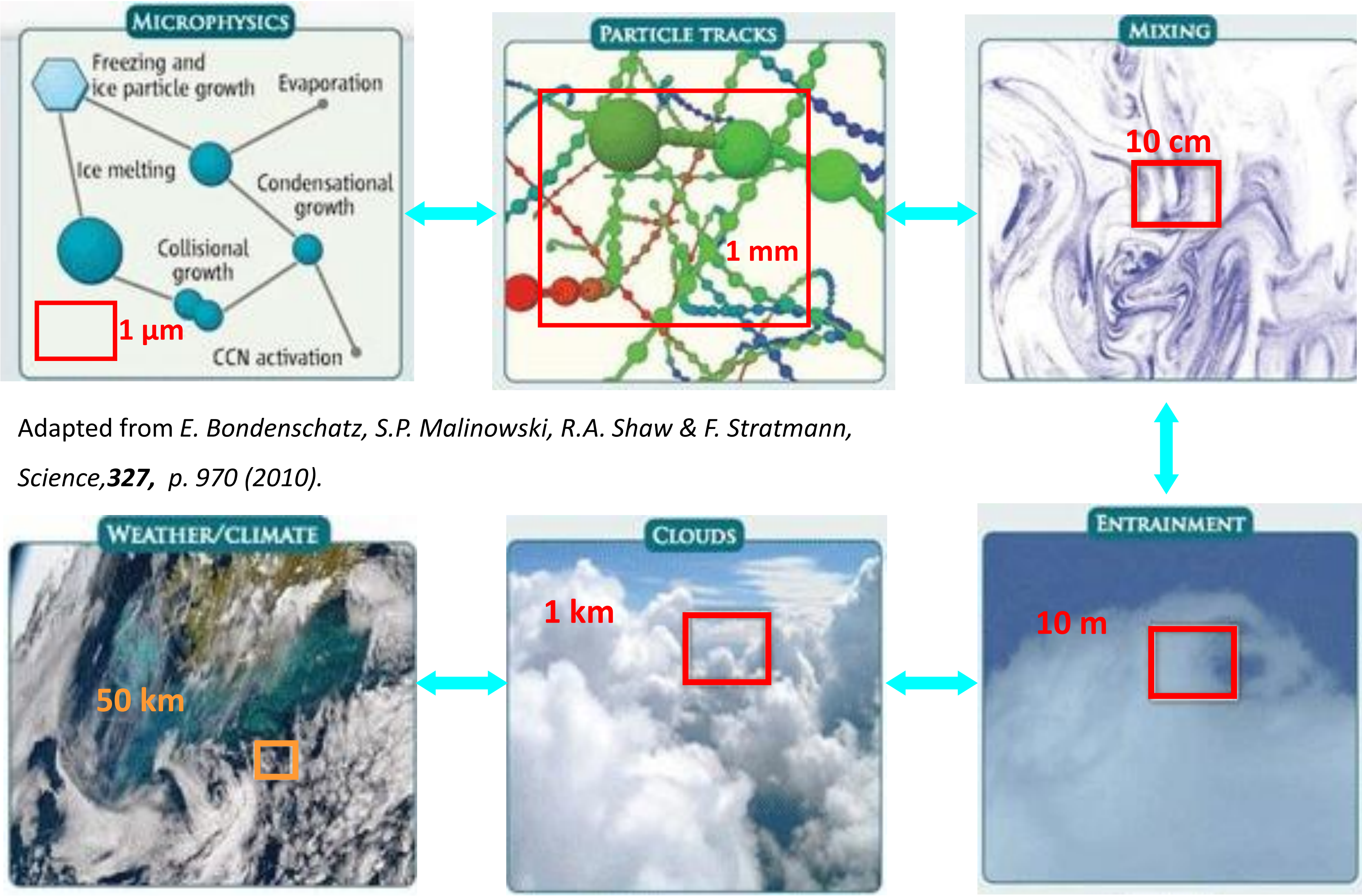
DNS for large domains: Challenges for computation and storage

Neethi Suresh[1], Lois Thomas[1], Bipin Kumar[1]
Ravi S. Nanjundiah[1,2] and Suryachandra A. Rao[1]

[1] Indian Institute of Tropical Meteorology, Pune, India

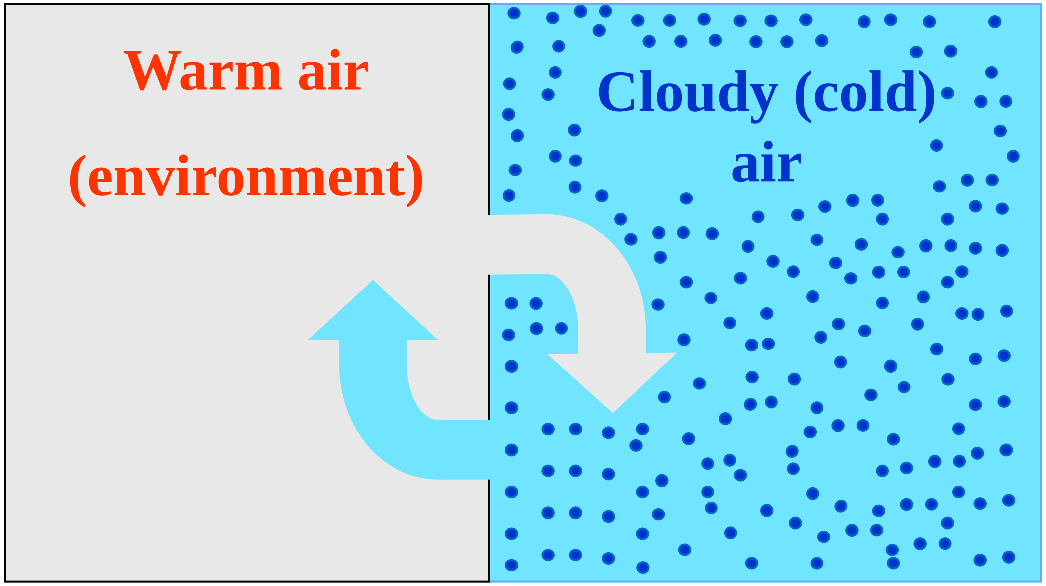
[2] Center for Atmospheric and Oceanic Science,
Indian Institute of Science, Bangalore, India

Motivation



Adapted from *E. Bondenschatz, S.P. Malinowski, R.A. Shaw & F. Stratmann, Science,**327**, p. 970 (2010).*

Motivation



Model

Eulerian (Fluid flow equations)

$$\nabla \cdot \mathbf{u} = 0$$

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho_0}\nabla p + \nu \nabla^2 \mathbf{u} + g\left[\frac{T - T_0}{T_0} + \epsilon(q_v - q_{v0}) - q_l\right]\vec{e}_z + f_{LS}$$

$$\partial_t q_v + (\mathbf{u} \cdot \nabla) q_v = D\nabla^2 q_v - C_d$$

$$\epsilon = \frac{R_v}{R_d} - 1$$

$$\partial_t T + (\vec{u} . \nabla) T = \kappa \nabla^2 \vec{u} + \frac{L}{c_p} C_d$$

κ : thermal conductivity

Periodic BC

R_v : vapor gas constant

R_d : dry air gas constant

q_v : vapor mixing ratio

q_l : liquid water content

f_{LS}: turbulent forcing (form large scale)

Lagrangian (droplet movement equations)

$$\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} = \frac{1}{\tau_p}[\mathbf{u}(\mathbf{X}, t) - \mathbf{V}(\mathbf{X}, t)] + \mathbf{g}$$

$$r(\mathbf{X}, t)\frac{\mathrm{d}r(\mathbf{X}, t)}{\mathrm{d}t} = KS(\mathbf{X}, t)$$

$$\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} = \mathbf{V}(\mathbf{X}, t)$$

$$S(\mathbf{X}, t) = \frac{q_v(\mathbf{x}, t)}{q_{v,s}} - 1$$

$$\tau_p = \frac{2\rho_l r^2}{9\rho_0 \nu}$$

Finite particle response time

ρ_l : water density

ρ_0 : air density

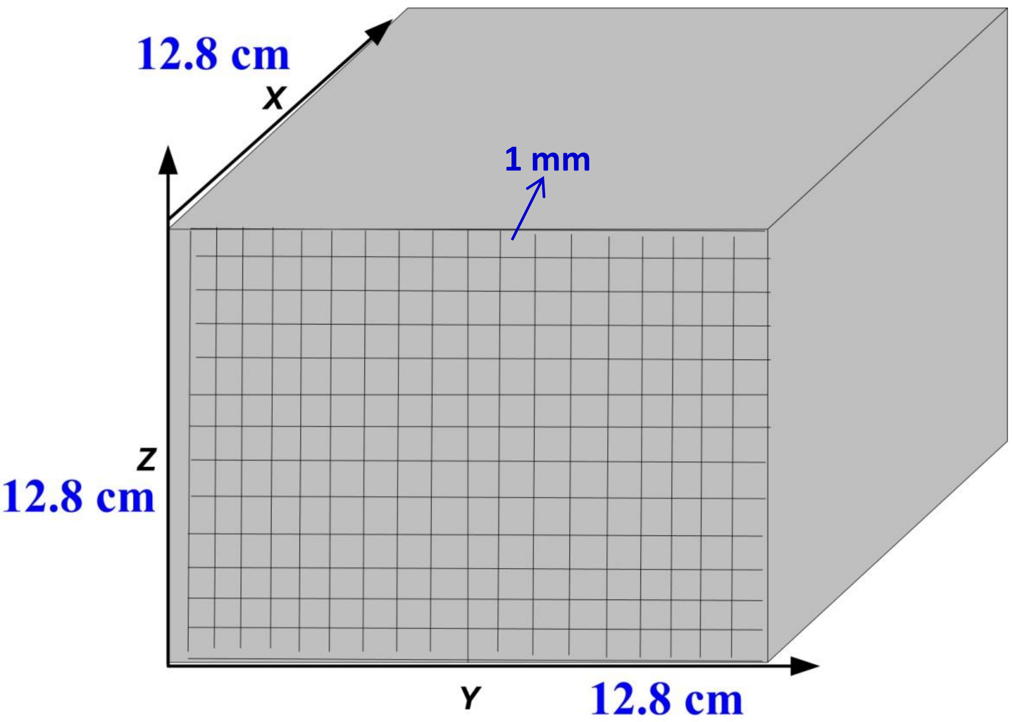
ν : kinematic viscosity

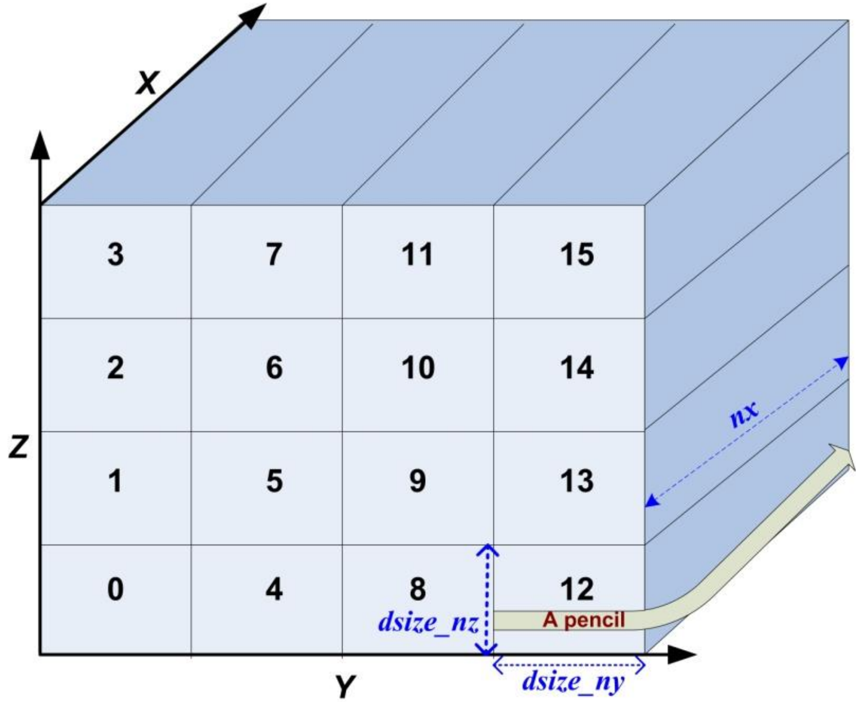
Kumar et al., *JAS, 2014, JAMES, 2017, Götzfried et al., JFM 2017*

Computational Details

Domain decomposition on 2D processor topology

- Pseudo-spectral method used to convert Partial Differential Equations (PDE) to set of Ordinary Differential Equations (ODE).
- System ODE is solved by 2nd order Predictor-corrector (time stepping) method.



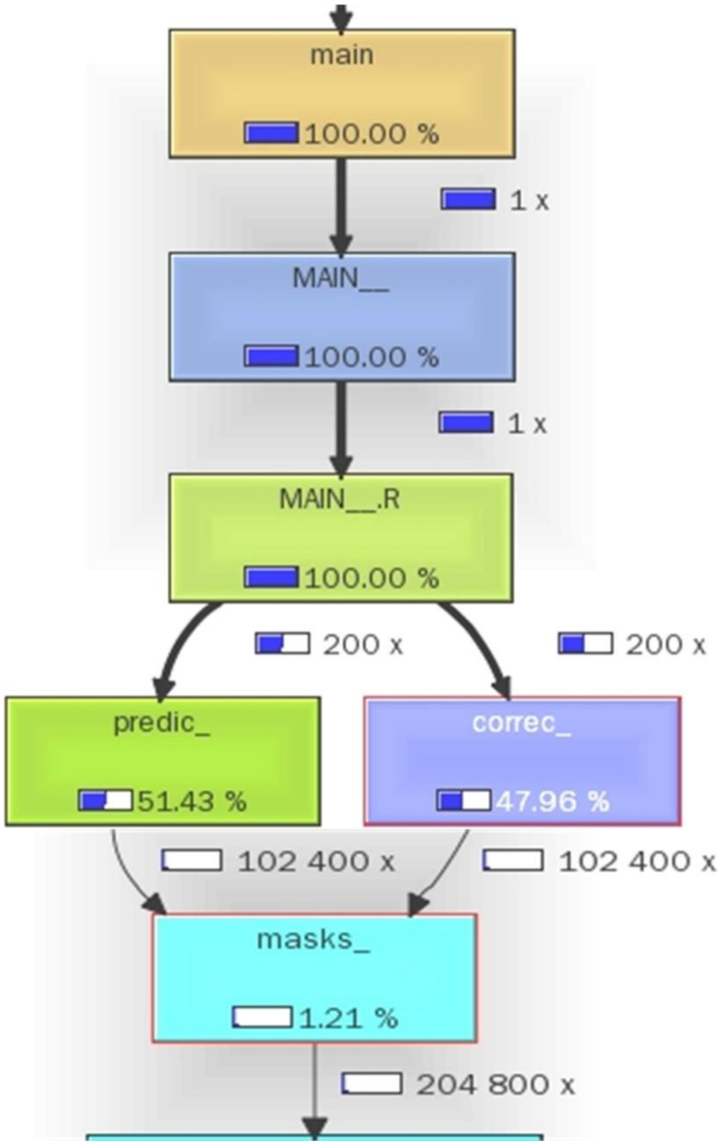


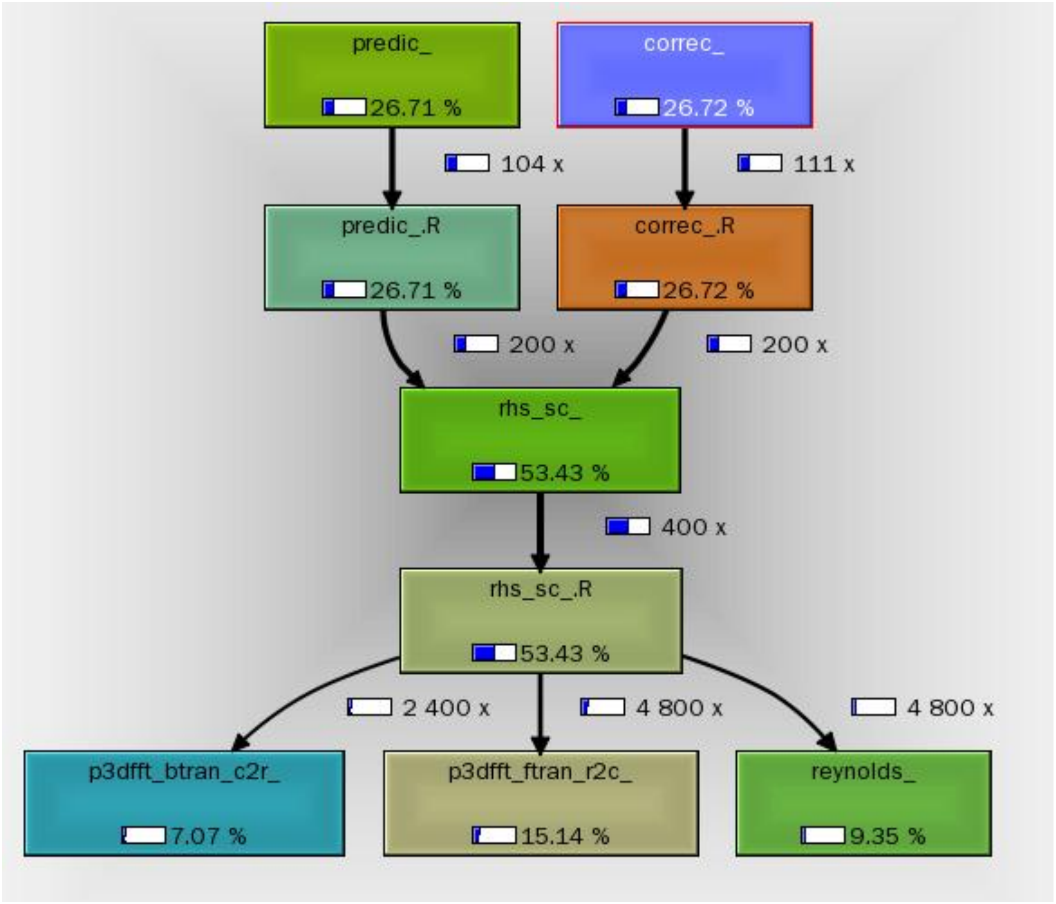
FORTRAN 90 + MPI + OpenMP

Flowchart

FORTRAN 90 + MPI + OpenMP

Code optimization: Profiling (DNS coding)

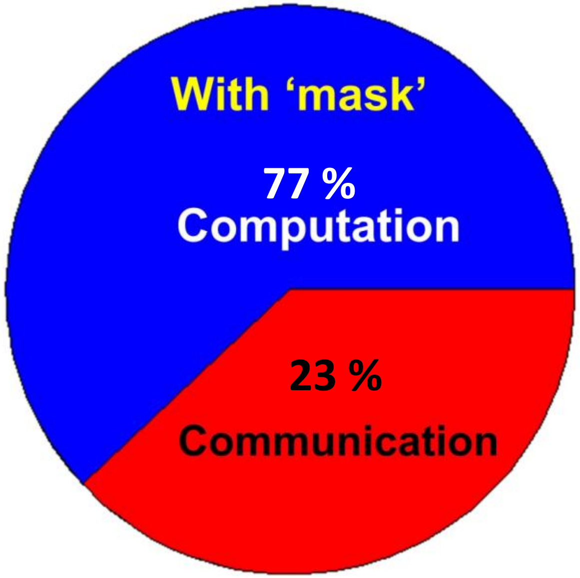




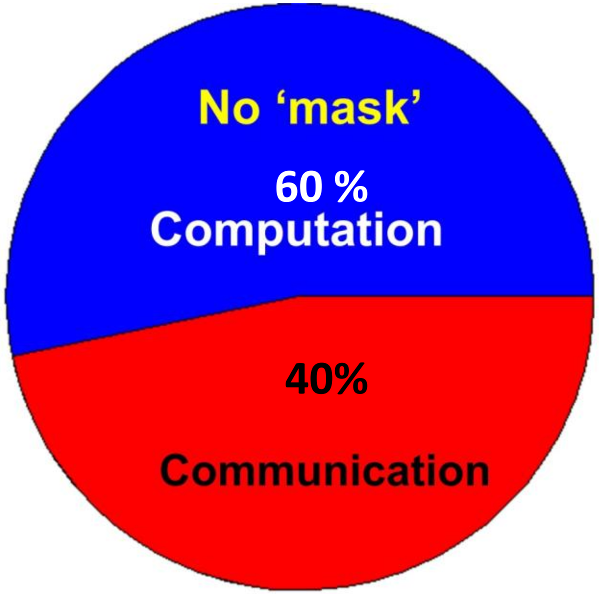
Code optimization

modification in source code

(cache-friendly code)



Before



After

Computational time reduced by 17%.

Computational details

Four different domains

D1 : 12.8 cm^3, D2: 25.6 cm^3,
D3: 51.2 cm^3 , D4: 102.4 cm^3

Total times for 60000 iteration using 1024 cores.

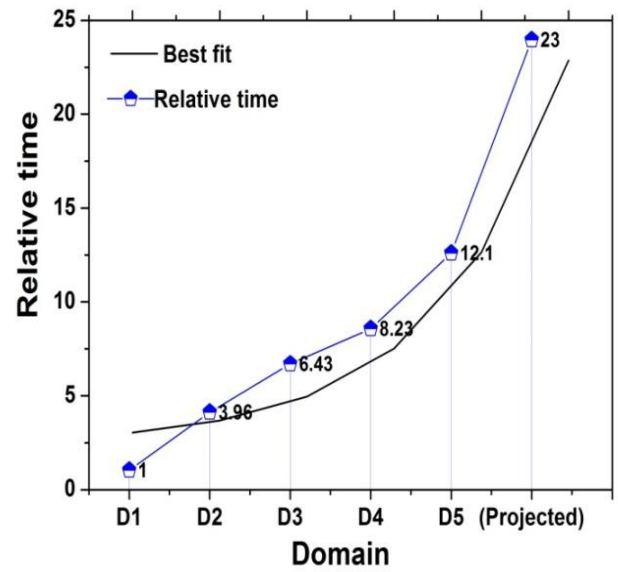
Domain (cm^3)	Time (sec)	Diff1	Diff2	N_t
12.8 | D1	2759	1	1	108134
25.6 | D2	10947	4	4	865075
51.2 | D3	70394	6.43	25.5	6920601
102.4 | D4	579642	8.23	210	57378078
204.8 | D5	6989561	12.1	2533	433166745

`Diff1' is the difference in times with respect to previous small domain. Similarly, difference in the times with respect to smallest domain is represented by `Diff2'.
N_t → #droplets required for that computational domain.

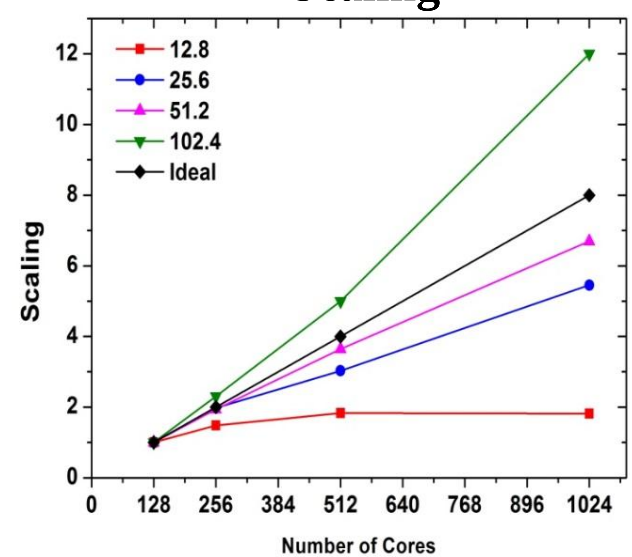
6989561 sec → 80 days : 2m^3

Requirement (for one experiment)

- Core: min (16384)
- Up to 500 TB
- More memory per core/node
- Very fast inter-processor connections



Scaling



Conclusions

- Carried out simulation of cloud micro-physics in different domains.
- Cope optimization saved 17% total time.
- Scaling analysis shows a superliner speed-up. Also optimized number of cores have been suggested for each domain.
- Provide projected time and required resources for bigger domain.

Thank you for your attention