RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

METEO FRANCE

About benchmarking with ~~toy~~ dwarf models

Ryad El Khatib / Météo-France
6th ACCORD ASW
Marrakech 13 - 17 April 2026

Plan

- General considerations
- Design of the dwarf model
- Case of Arome-France and Arpege on Météo-France HPC (operational and benchmark resolutions)
- « Universal » dwarf model ?
- Conclusion

Introduction

- Benchmarking an application in a realistic way requires that it runs at its full resolution ...
- … or even at higher resolution to prepare an increase of resolution or a change of HPC.

- HPC vendors are requesters of smaller applications in order to fine-tune theirs offers (they need to run many tests !)

- => this is a very expensive activity for everybody !
  (Ideally we wish to use a single-node model)

- But still, smaller models would not reflect properly I/O footprint or scalability/network (at least)

Can we find a compromise ?

Let's be pragmatic

- Forget about I/Os :
  - ➔ We may design a full-size I/O server (ioserv executable) coupled to a dummy model

- Forget *partly* about scalability and network :
  - ➔ Local communications (like SL) could be benchmarked
  - ➔ But no way for all-to-all communications (like in spectral space)

- Focus on compute-bound and memory-bound aspects
  - ➔ For us users : to the benefit of cpu/memory optimizations
  - ➔ For HPC vendors : to select the best chip an the best memory characteristics

Other issues to design dwarf models

- Is the memory cost per task constant or almost constant from the full resolution to the dwarf model resolution ?
  => To be verified by experiments.

- Nodes can have different sizes from one architecture to another.
  Exemples of EuroHPC :
  - Leonardo (Intel chips) 2x56 cores, 512 GB
  - Lumi-C (AMD chips) 2x64 cores, 256 GB
  - Jupiter (ARM chips) 2x80 cores, 512 (+128 HBM) GB

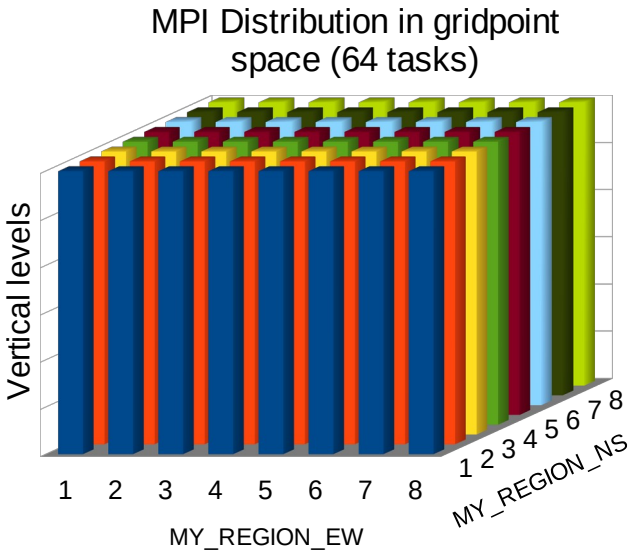
  => Can we design a dwarf model
  fairly equal for all machines ?

Design of a dwarf model for a given computer

- The dwarf model should look like an extraction of MPI tasks from the real model
- In order to keep the surface more or less realistic (proportion of land, sea, etc) the horizontal domain should be the same

  => the horizontal resolution must be smaller

  => The lower resolution must not impact other dimensions, such as the Semi-Lagrangian halos



Source:Wikimedia Commons

Design of the dwarf model in gridpoint space

Convention : X = value in the real model ; X' = value in the dwarf

N = Number of nodes ; *t* = number of tasks per node

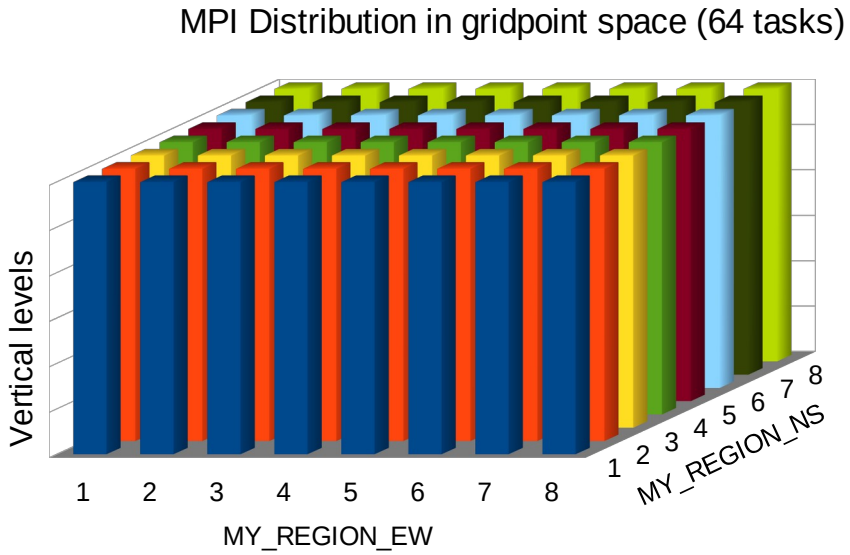
- **Number of vertical levels NFLEVG should be equal**
- **Number of gridpoints per task NGPTOT should be equal**

  NGPTOT = NGPTOTG/NPROC
  ≈ k*NDLON*NDGLG/NPROC
  ≈ k*NDLON²/NPROC
  ≈ k*NDLON²/(*N***t*)

- ***Actual* halo width should be equal**

  NSLWIDE(x) = V * Δt / Δx
  ≈ k*NDLON

NFLEVG'=NFLEVG
NDLON'=NDLON / √((*N*t*)/(*N'***t'*))**
NDGLG'=NDGLG / √((*N*t*)/(*N'***t'*))**
NSLWIDE'=NSLWIDE * √((*N*t*)/(*N'***t'*))**

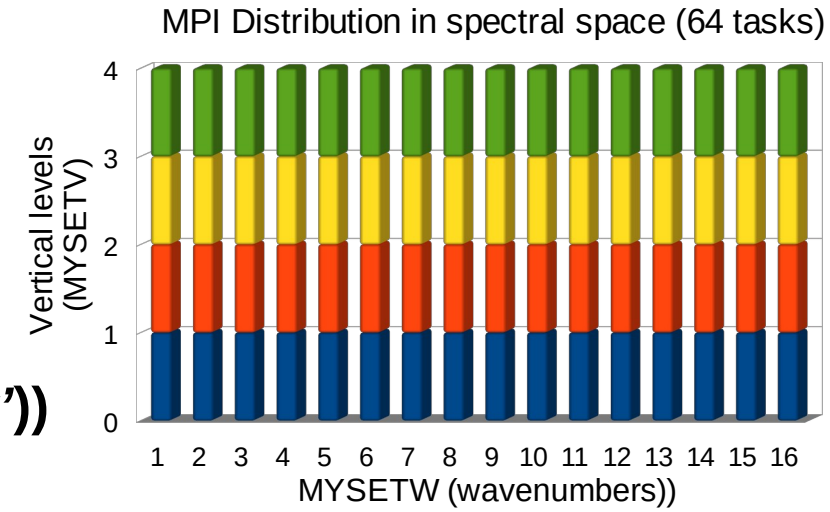


Design of the dwarf model in spectral space

N = Number of nodes ; *t* = number of tasks per node

- **Number of vertical levels per task NFLEVL should be equal**
  NFLEVL ≈ NFLEVG/NPRTRV => NPRTRV should be equal
- **Number of wavenumbers per task NSPEC2 should be equal**
  NSPEC2 ≈ NSPEC2G/NPROC
  = k*(NSMAX+1)*(NMSMAX+1)/NPROC
  ≈ k*(NSMAX+1)²/NPROC
  ≈ k*(NSMAX+1)²/(*N***t*)

NPRTRV'=NPRTRV
NSMAX'+1=(NSMAX+1) / √((*N*t*)/(*N'***t'*))**
NMSMAX'+1=(NMSMAX+1) / √((*N*t*)/(*N'***t'*))**



Design of the dwarf model: 2-ways distribution constraints

- (for LAM model or global model with LEQ_REGIONS=F)
  To preserve the horizontal shape of the tasks in gridpoint space:
  NPRGPNS'/NPRGPEW' = NPRGPNS/NPRGPEW = r
- (as seen before)
  To preserve the vertical distribution in spectral space:
  NPRTRV'=NPRTRV=v
- Knowing that :
  NPROC=NPRGPNS*NPRGPEW=N*t
  NPROC=NPRTRW*NPRTRV=N*t

Finally:

NPRGPNS'=√(r*N'*t')
NPRGPEW'=N'*t'/NPRGPNS'
NPRTRW'=N'*t'/v
NPRTRV'=v

And (because of rounded values) check that :
NPRGPNS'*NPRGPEW'=NPRTRV'*NPRTRW'=N'*t'

Design of the dwarf model: Handling of the extension zone

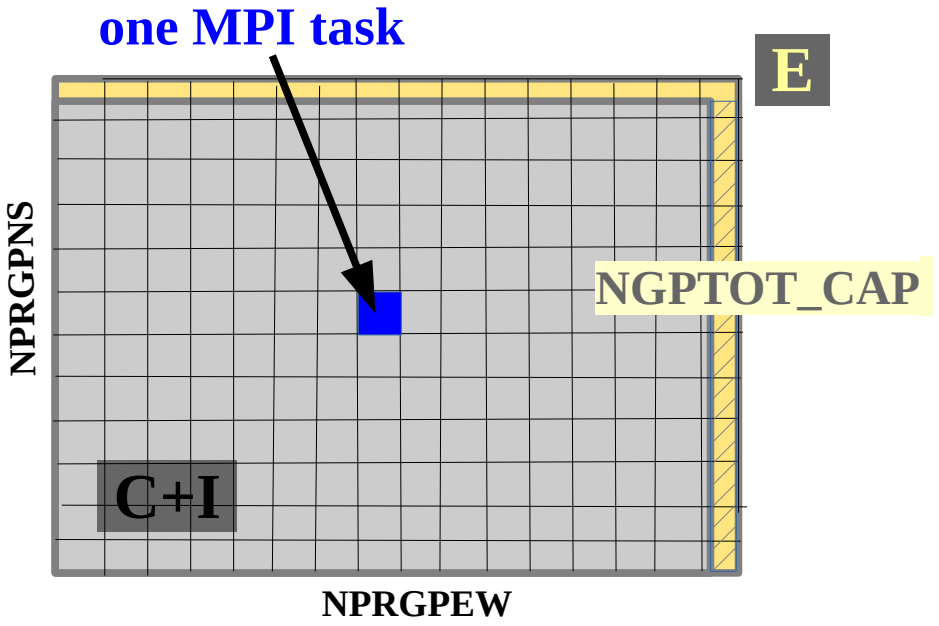
- **Width of E-zone = 11 gridpoints** < NDLONL or NDGLL in practice
- In the distributed model it may impact only the load balancing, not much the speed
- **Percentage of tasks which size are reduced by the E-zone :**

  For a square distribution : e = (NPRGPEW)/NPROC ≈ 1 /√(N*t)

  For a 1-way distribution (« A level ») : e = 1/(N*t)

=> the smaller the dwarf will be, the more the E-zone may affect the load balancing

Solution: juste ignore the E-zone : Compute over E thanks to the key **LGPTOT_CAP=.FALSE.**

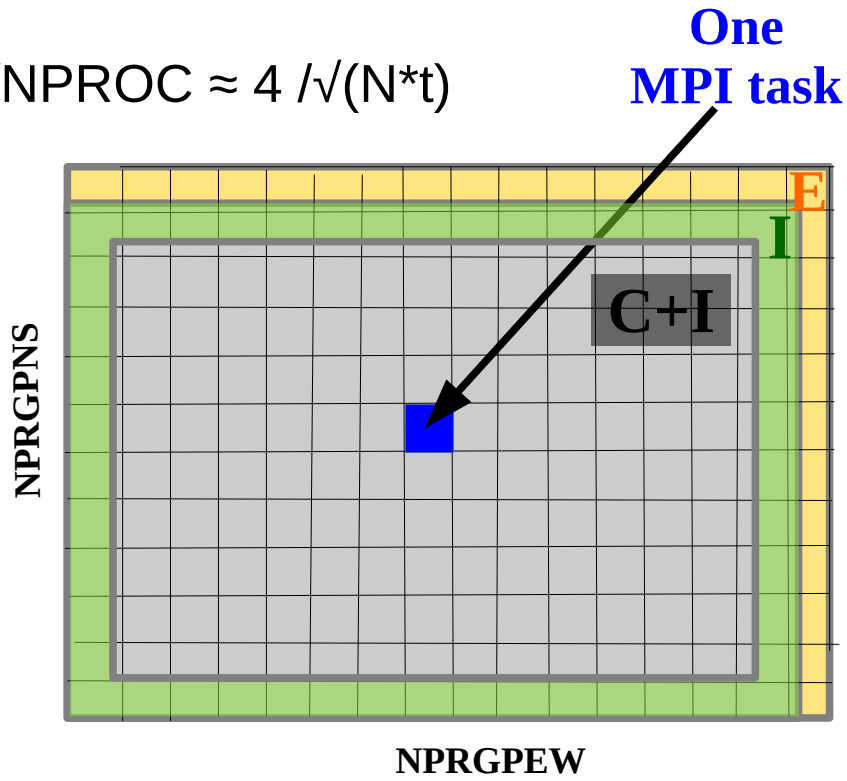


Design of the dwarf model: Complication with the coupling zone

- **Width of I-zone in the real model : 20 km**
  ≈ 8 to 16 gridpoints < NDLONL or NDGLL in practice
  => To be kept in the dwarf as it should be like a subset of tasks from the real model
- **Percentage of tasks computing coupling :**
  For a square distribution :
  c = (2*NPRGPNS+2*NPRGPEW-4)/NPROC ≈ 4 /√(N*t)
  For a 1-way distribution (« A level ») :
  c = N*t

=> the smaller the dwarf will be, the more expensive the coupling will seem

Workaround :
reduce the coupling frequency proportionally ?



Case of Arome-France today

Searching for dwarves with the same distribution ratios

	Arome el700	Dwarf #1	Dwarf #2	Dwarf #3
N (number of nodes on Belenos)	18	1	2	8
NDLON x NDGLG	1440 x 1536	339 x 362	480 x 512	960 x 1024
NFLEVG	90	90	90	90
NMSMAX x NSMAX	719 x 767	169 x 180	239 x 255	479 x 511
NSLWIDE extension factor	1	4,24	3,00	1,50
NPRGPNS x NPRGPEW	24 x 24	**8 x 4**	8 x 8	16 x 16
NPRTRW x NPRTRV	72 x 8	4 x 8	8 x 8	32 x 8
r (shape of gridpoint tasks)	1	**2**	1	1
t (tasks per node)	32	32	32	32
ratio of tasks reduced by E-zone	4,17 **%**	**12,50** %	**12,50** %	6,25 %
ratio of coupling tasks	16 %	63 %	44 %	23 %
(rounded) coupling frequency (h)	1	4	3	2
Mesh size	1,3 km	**5,7 km**	**4,0 km**	2,0 km
Billing units reduction factor	1	18	9	2,25

(One node looks not enough)
A reduction by a factor of 9 could make a cheap meso-scale dwarf
A « grey-zone » dwarf requires a reduction by a factor of 2.25 only

Case of Arome-France in benchmark procurement

Searching for dwarves the same distribution ratios

	Arome ec650	Dwarf #1	Dwarf #2	Dwarf #3
N (number of nodes on Belenos)	98	1	6	24
NDLON x NDGLG	2500 x 2700	253 x 273	619 x 668	1237 x 1336
NFLEVG	120	120	120	120
NMSMAX x NSMAX	624 x 674	62 x 67	154 x 166	308 x 333
NSLWIDE extension factor	1	9,90	4,04	2,02
NPRGPNS x NPRGPEW	98 x 16	**8 x 2**	24 x 4	48 x 8
NPRTRW x NPRTRV	98 x 16	**1** x 16	6 x 16	24 x 16
r (shape of gridpoint tasks)	6,125	**4**	6,125	6,125
t (tasks per node)	16	16	16	16
ratio of tasks reduced by E-zone	1,02 **%**	**12,50** %	4,17 %	2,08 %
ratio of coupling tasks	14 %	100 %	54 %	28 %
(rounded) Coupling frequency (h)	1	7	4	2
Mesh size	750 m	**7,7 km**	**3,1 km**	1,5 km
Billing units reduction factor	1	98	16	4

(One node looks not enough)
A reduction by a factor of 16 should make a cheap meso-scale dwarf
A « grey-zone » dwarf requires a reduction by a factor of 4 only

Case of Arpege today

Searching for dwarves the same distribution ratios

	Arpege tl1798	Dwarf #1	Dwarf # 2
N (number of nodes on Belenos)	27	1	3
NDLON x NDGLG	3600 x 1800	693 x 346	1200 x 600
NFLEVG	105	105	105
NSMAX	1798	345	599
NSLWIDE extension factor	1,00	5,30	3,00
NPRGPNS x NPRGPEW	24 x 18	**6 x 3**	8
NPRTRW x NPRTRV	72 x 6	3 x 6	8 x 6
r (shape of gridpoint tasks)	1,33	**2**	1,33
t (tasks per node)	16	16	16
Billing units reduction factor	1	27	9

(One node looks not enough)
A reduction by a factor of 9 seems possible

Case of Arpege in benchmark procurement

Searching for dwarves the same distribution ratios

	Arpege tc1799	Dwarf #1	Dwarf #2	Dwarf #3
N (number of nodes on Belenos)	152	4	27	8
LEQ_REGIONS	F	**T**	F	**T**
NDLON x NDGLG	7200 x 3600	584 x 292	3072 x 1536	1652 x 826
NFLEVG	120	120	120	120
NSMAX	1799	145	767	412
NSLWIDE extension factor	1,00	12,33	2,34	4,36
NPRGPNS x NPRGPEW	38 x 32	-	16 x 14	-
NPRTRW x NPRTRV	38 x 32	**1** x 32	7 x 32	2 x 32
r (shape of gridpoint tasks)	1,19	1	1,19	1
t (tasks per node)	8	8	8	8
Billing units reduction factor	1	38	6	19

(One node is not enough)
A reduction by a factor of 6 seems possible

or a reduction by a factor of 19 with LEQ_REGIONS=TRUE
But with a different communication pattern in SL comms

Back to a « universal » dwarf model

- Easier to design because we are not bound to a specific MPI distribution
- But simulation of local communications will be more approximative
- The other rules can be applied :
  - Same number of vertical levels
  - NSLWIDE extension factor equal to the mesh reduction factor
  - Prefer LGPTOT_CAP=.FALSE.
  - Reduce coupling frequency to preserve the proportion of coupling tasks

=> focus on CPU and memory aspects preferably ?
=> make not too small models to preserv the physical modelisation … or reduce also the size of the domain

Conclusion

- Dwarf models can be designed for benchmarking, following certain rules discussed in this presentation
- The dimensions of a dwarf model are driven by the modelisation we want to study.
  Make dwarves, not toys !
- A dwarf model for a specific computer can be designed with more accuracy than for a non-specified computer
- All this is theory.
  Experiment is now needed to confirm it or not !

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

METEO FRANCE

Thank you for your attention