Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Eidgenössisches Departement des Innern EDI
Bundesamt für Meteorologie und Klimatologie MeteoSchweiz

Porting the COSMO nudging code to GPU using directives

Daniel Leuenberger and Tiziano Diamanti, MeteoSwiss

COSMO General Meeting, Lugano, 10.9.2012

Motivation

Suppose you would be able to do your daily COSMO simulations on a computer which

- is a factor of 7 cheaper
- uses only ¼ of the electrical power (and thus saves maintenance costs)

in the same time to solution!

Next-generation Graphical Processing Units (GPU) based architectures could make this dream come true

Need to make COSMO fit for Graphical Processing Unit (GPU) architectures

Swiss HP2C Project

- COSMO code is memory bandwith limited (only 2% peak performance on our CRAY XE6)
- GPU has much higher memory bandwith than CPU
- Adapt COSMO to run efficiently on mixed CPU-GPU computers
- Rewrite of the dynamical core in C++ using a newly developed stencil library
- Use compiler directives to port physical parametrizations (does not change the fortran code)
- COSMO PP POMPA

OPCODE Project

- Bring current MeteoSwiss operational COSMO production to a new CPU-GPU demonstrator hardware to achieve same time to solution
- Need to consider also nudging data assimilation!

OPCODE Demonstrator

144 CPU
(1728 cores)

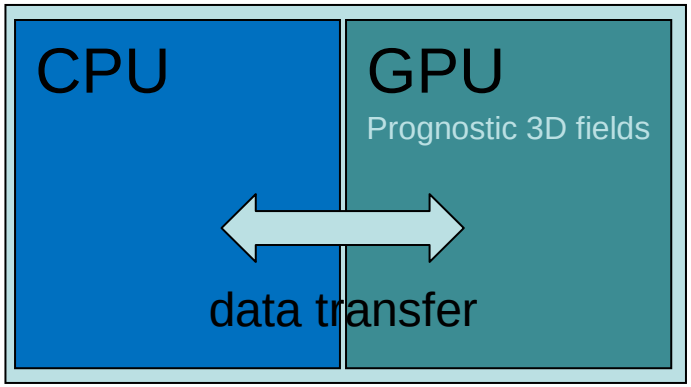
16 GPU and
2 CPU (16 cores)

Cray XE6 (3 cabinets)

OPCODE Demonstrator

Porting Strategy

- Use GPU for computing intensive parts (dynamics, physics)
- Use CPU for I/O and less computing intensive parts
- Avoid CPU-GPU data transfer whereever possible (expensive!), keep prognostic 3D fields on GPU!



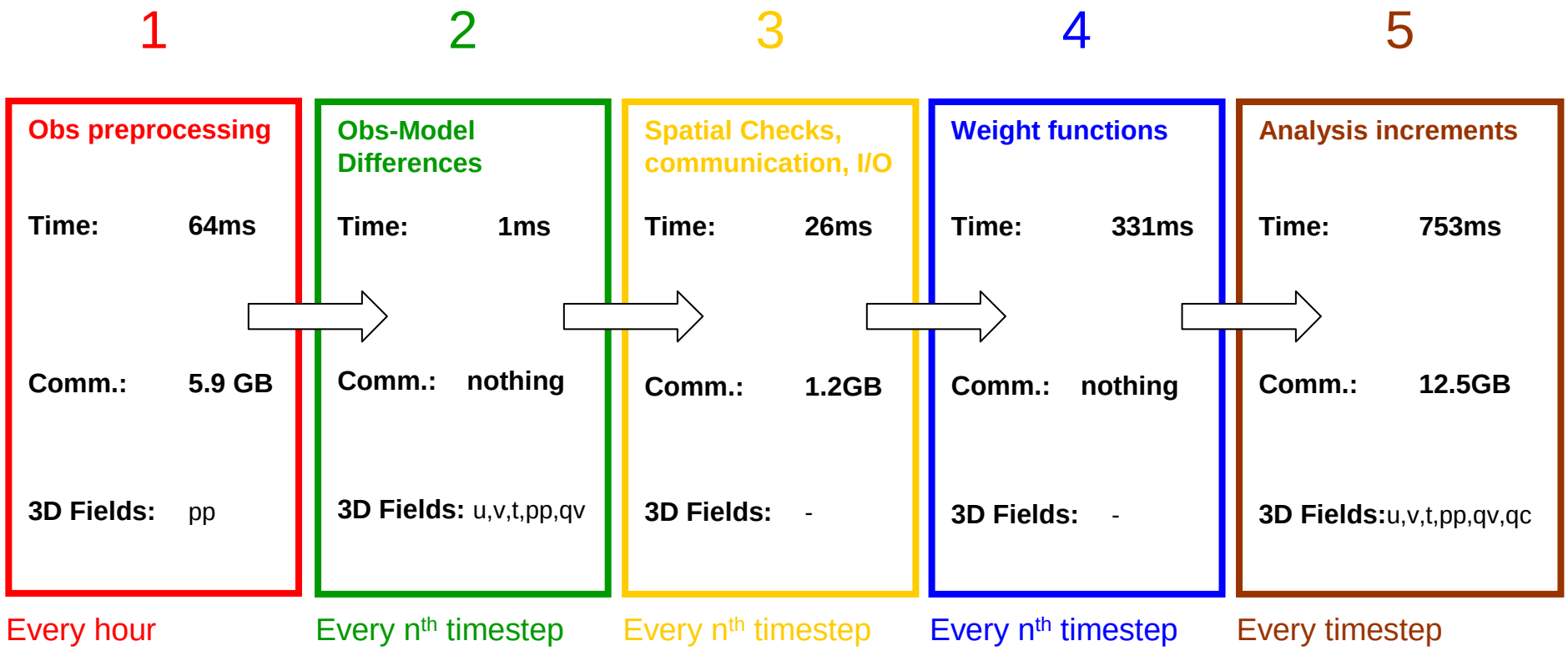
- Assimilation code is very large (ca 83’000 lines of code, ca 37% of whole COSMO model)
  - Use directives to port code
  - Only port code parts to GPU which involve prognostic 3D fields

Example use of directives


```
do j=1,Ny
   do i=1,Nx
      a(i,j)=b(i,j)*c(i,j)
   end do
end do
```

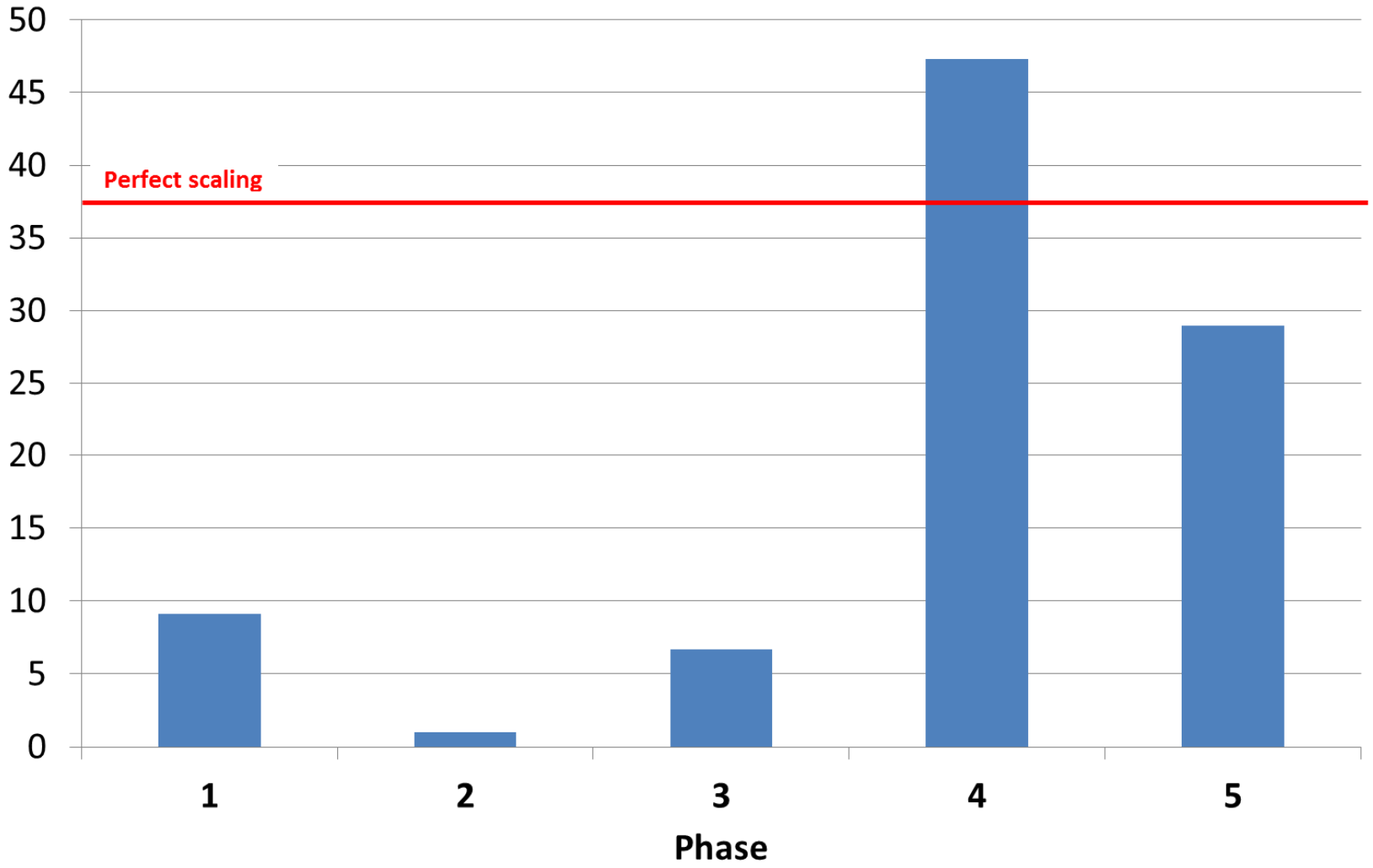
```
!$acc data create(a,b,c)
!$acc update device(b,c)
!$acc parallel
do j=1,Ny
   do i=1,Nx
      a(i,j)=b(i,j)*c(i,j)
   end do
end do
!$acc end parallel
!$acc update host(a)
!$acc end data
```

Analysis of Nudging Code



Scaling of Nudging Code

Ratio of execution time 16 vs 600PE



Results for Phase 5

- Ported whole phase, including the subroutines
  - geostroph_ps_corr, ps_temperatur_corr, nudge_humid_mass and nudge_horiz_wind
- Verified results: same results up to machine accuracy
- Performance of GPU version compared to CPU (16cores) version
  - 2 times slower with our porting (DA phase 5 on GPU, rest on CPU)
  - 160 times slower without our porting (whole DA on CPU, rest of the model on GPU, needs data transfer of prog. fields)
- Goal achieved: avoided data transfers between CPU and GPU

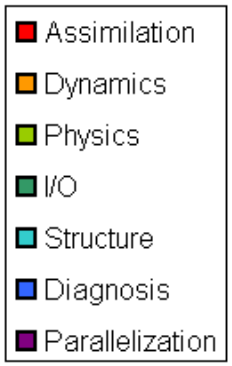
Further Remarks

- Started to implement a mixed CPU-GPU version of the nudging code
- Expected speed-up of other model parts:
  - Dynamics: factor 6
  - Physics: factor 3
- DA is switched on only during ca 12% of our COSMO simulations, so a slower DA performance on the GPU is not critical
- Current results indicate that time to solution will be similar on the GPU hardware and on our operational CRAY hardware

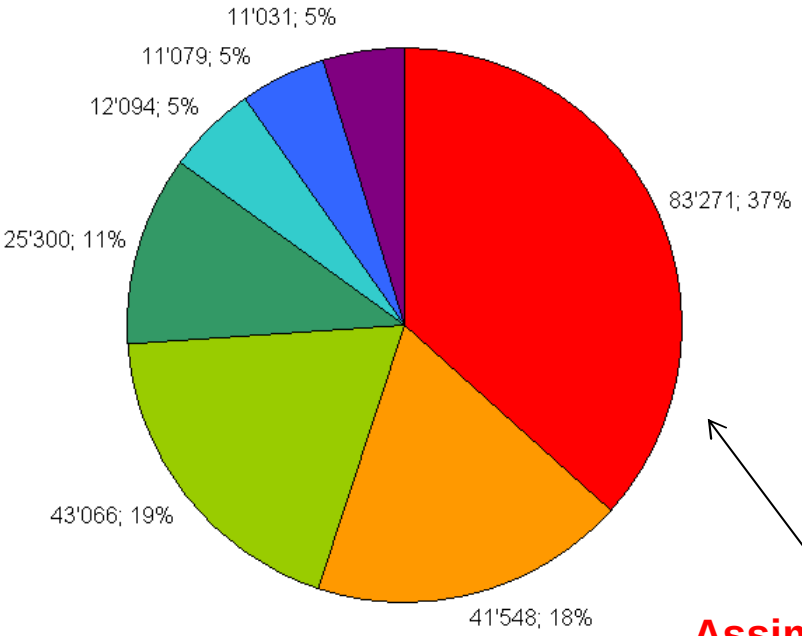
Thanks for your attention

Lines vs. Runtime

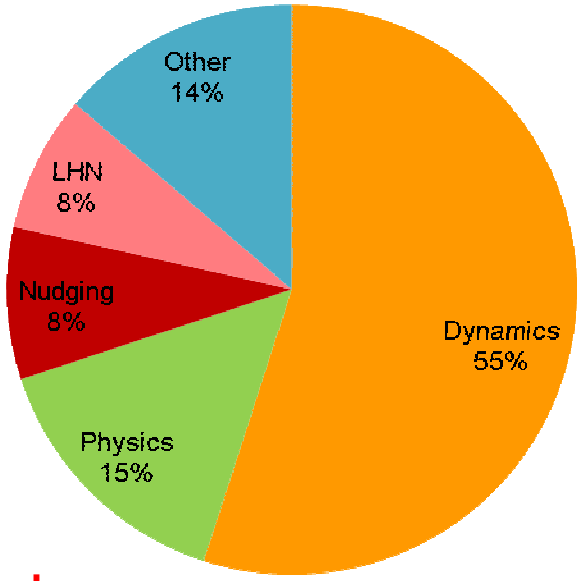
- 250’000 lines of Fortran 90 code



% Code Lines



% Run time



Assimilation mode:
16% run time
37% of code lines

Phases' Analysis

	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5
1) Total time	554s	8.9s	231s	2860s	6506s
2) Number of times the phase is executed	1 per hour, 3 times in hour suite	Every 12th step (540 / 12)	Every 12th step (540 / 12)	Every 12th step (540 / 12)	Every step 540
3) Total time per call	184.7s	0.2s	5.13s	65.5s	12s
4) Total time per call per cpu	11.5s	0.012s	0.321s	3.97s	0.753s
5) Time per step per cpu	0.064s	0.001s	0.0268s	0.331s	0.753s

Meaning of the rows:

1)Total time is the time measured by Scalasca, adding all the procedures belonging to a certain phase.
2)Number of times the phase is executed: only phase 5 is executed every time step, the others are not (we considered Cosmo 2)
3)Total time per call: 1) divided by 2)
4)3) divided by number of processes
5)1) divided by num. time steps (540 in this case) divided by number of processes

Porting effort

Phase 5		
Function name	**Subfunctions**	**Number of lines**
nudge_humid_mass		1100
	Satad	200
	Get_gs_lheating	100
nudge_horiz_wind		1500
Phase 4		
ps_spreading		500
mult_org_spread		700
upair_org_spread		600
surf_org_spread		500
Phase 2		
local_sort_reports		1400
ps_local_info		750
upair_local_info		800
surf_local_info		800
mult_org_localinfo		1300
	mult_vertic_intpol	2300
	mult_obs_increment	900

Nudging: Possible subdivision between CPU and GPU (2)