Optimizations of CANARI

Robin Degrave (ENSEIRB), Ryad El Khatib,
Françoise Taillefer, Philippe Marguinaud (CNRM/GMAP)

22nd Aladin Workshop & Hirlam all staff meeting
Marrakech 07-10 May 2012

METEO FRANCE
Toujours un temps d'avance

Why this talk ??

- Rumors :
  - *Further optimization of Canari on scalar platform is needless :-)*
  - *Further optimization of Canari on vector platform is hopeless :-(*
- Directors :
  - *Must Canari be included in the benchmark of Arome ?*
- *Survey of any operationnal – or widely used - application to be monitored regularly*
- *Can we optimize the distribution of computing ressources between Canari and 3DVar for a full assimilation suite ?*

Plan

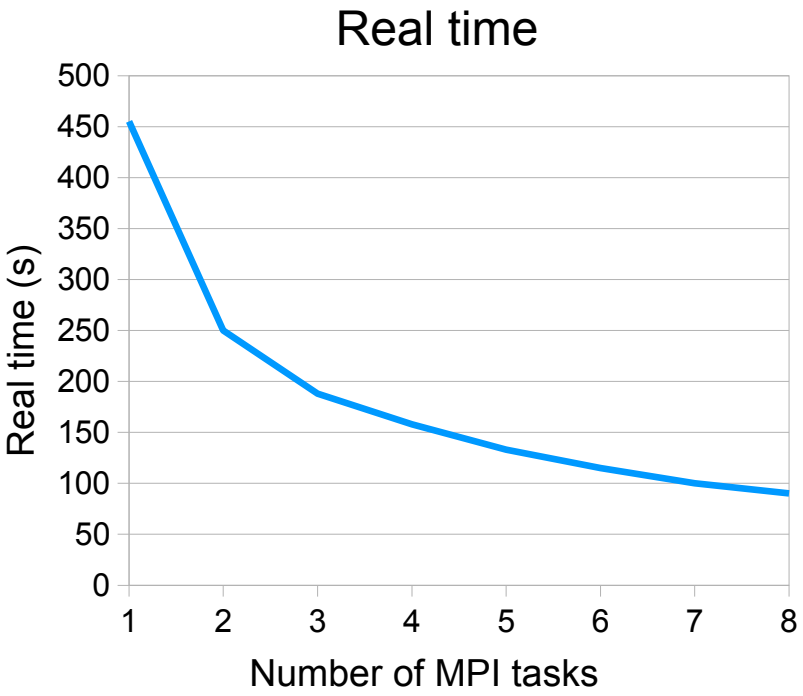
Optimization work

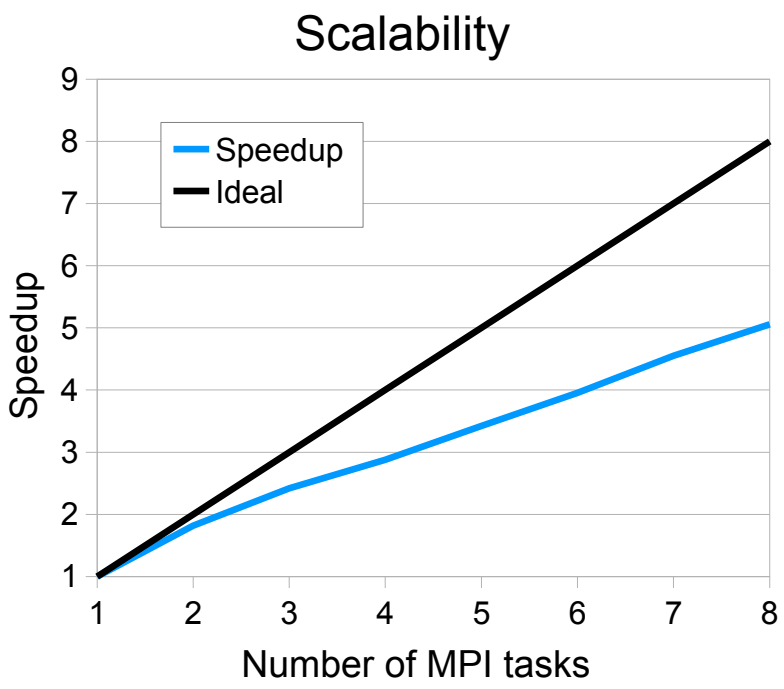
- On scalar platforms
- On vector platforms
- Integration aspects

Conclusion

CANARI on scalar(1) platforms

(Canari-Arome-France : 750 x 720 gridpoints, 60 levels)





- **Scalability : not very good**
- **Memory cost : excessive** : 9 Gb (1 proc) to 12 Gb (8 procs)

(1) Intel Westmere + Intel compiler

CANARI on scalar platforms

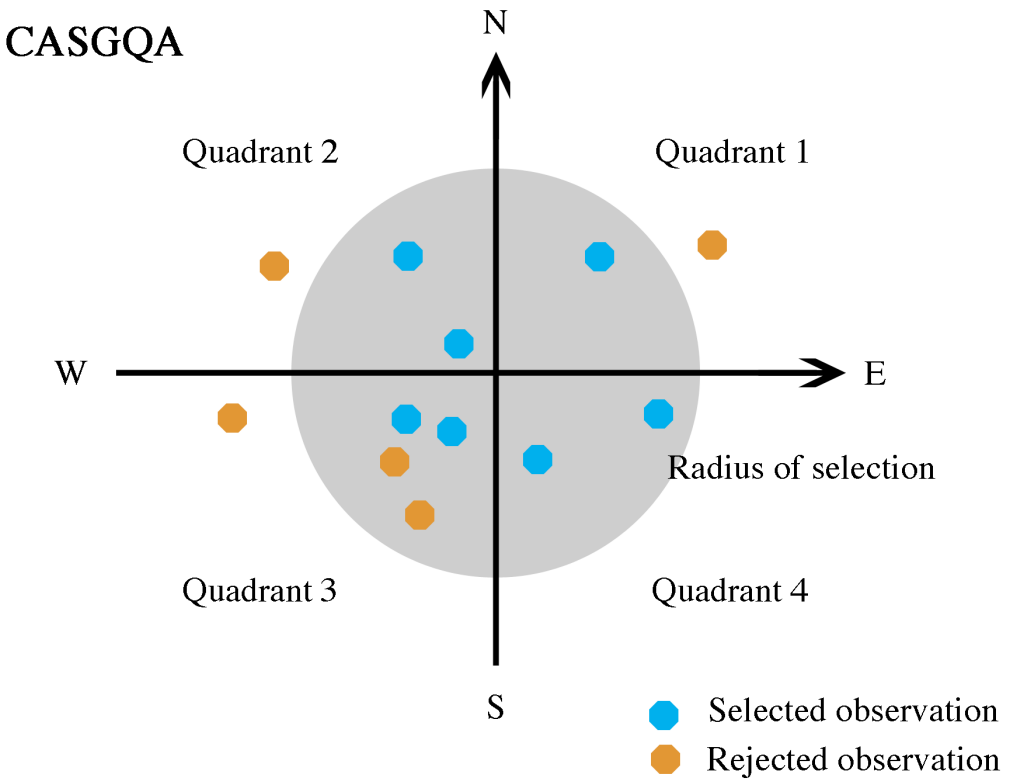
DrHook profiling for 2 MPI tasks

Avg-(%)	Avg.time (s)	Min.time (s)	Max.time (s)	St.dev (s)	Imbalance (%)	Calls (#)	Name
35.30	84.52	73.24	95.80	15.95	23.54	1051806	CASGQA
7.45	17.84	0.69	34.99	24.26	98.04	48	TRGTOL
11.84	28.35	26.07	30.63	3.23	14.91	1051806	CASGRA
7.74	18.54	17.46	19.62	1.53	11.03	1577772	CASPIA
3.37	8.08	7.65	8.50	0.60	10.02	1512166	MINV:GECO

=> shows up **performance anomalies** :

- ≈ 35% of time spent in a sorting algorithm (CASGQA)
- ≈ 7.5% of time spent in useless spectral transpositions (TRGTOL)
- Much trigonometric computations (CASGRA)
- Many calls to subroutines

CANARI on scalar platforms



CASGQA :

Selection

in each quadrant

of a limited number

of observations at a limited distance

of the gridpoint

CANARI on scalar platforms

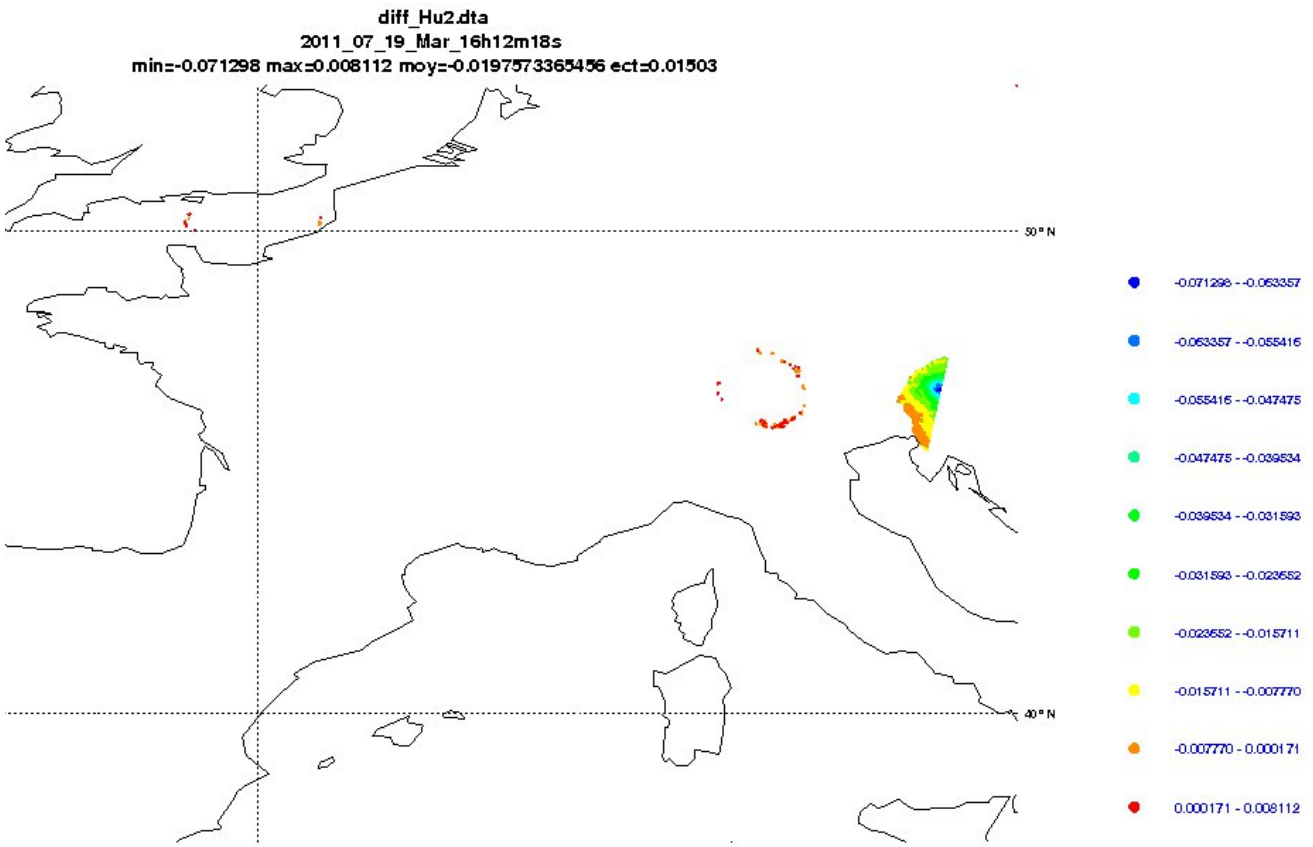
CASGQA :

the quadrant research

was incorrect, leading to slightly wrong results

(≈ 0.02%)

And excessive trigonometric computations

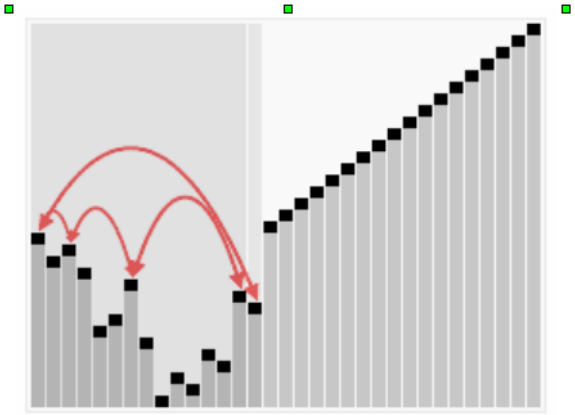


=> The correct formulation helps save 15% cpu time in CASGQA

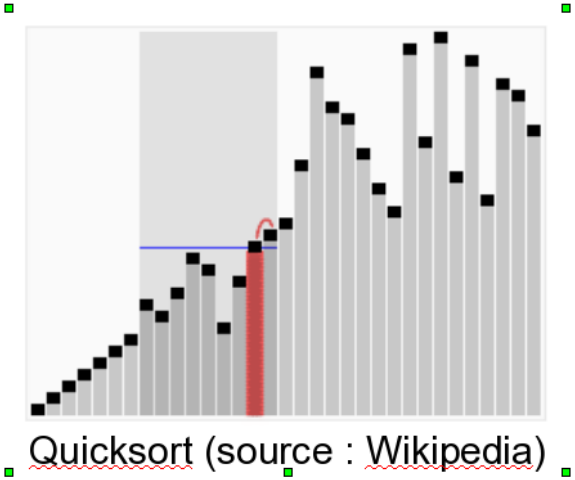
CANARI on scalar platforms

Research of a better performing sorting algorithm for CASGQA

- Initially : apparently a truncated algorithm in O(n2) (the first observations are not sorted)
- Fast algorithm are in O(nLog(n))
- Heapsort (used in ODB) has been disappointing (truncation lost)



Heapsort (source : Wikipedia)



Quicksort (source : Wikipedia)

Eventually

CASGQA was just a clumsily-coded Quicksort-like algorithm :

the pivot value had to be saved rather than recomputed

=> saves 60% cpu time in the sorting algorithm

CANARI on scalar platforms

Reduction of the memory cost :

=> A new option in the model, in order to handle only a subset of vertical levels :

- **&NAMCT0 : LIOLEVG=.FALSE.**
  - model reads only the NFLEVG lowest levels from file
  - supposes to set NFLEVG in namelist, too
  - I/Os are preserved (thanks to the indexed sequential property of the FA files)
  - Canari 3D algorithm is preserved
  - Can be used for post-processing as well (with care)

CANARI on scalar platforms

With LIOLEVG=.FALSE. in Canari,
we can set NFLEVG=2 despite the model has 60 levels

- Memory usage falls down to ≈ 1 Gb for 1 core
  88 % memory saved

- We perform minimal I/Os, setup and spectral transforms
  12 % cpu time saved

- Miscellaneous other optimisations :
  ≈ 3 % cpu time saved

CANARI on scalar platforms

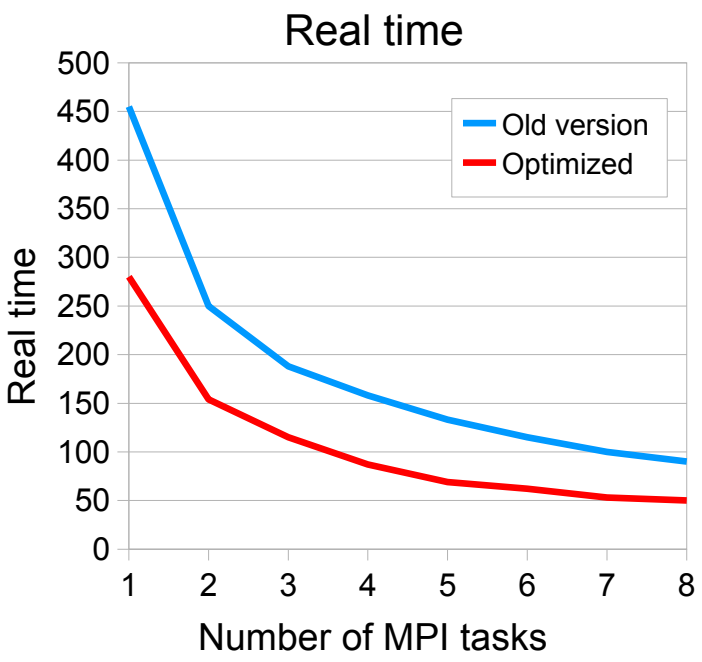
New DrHook profiling for 2 MPI tasks

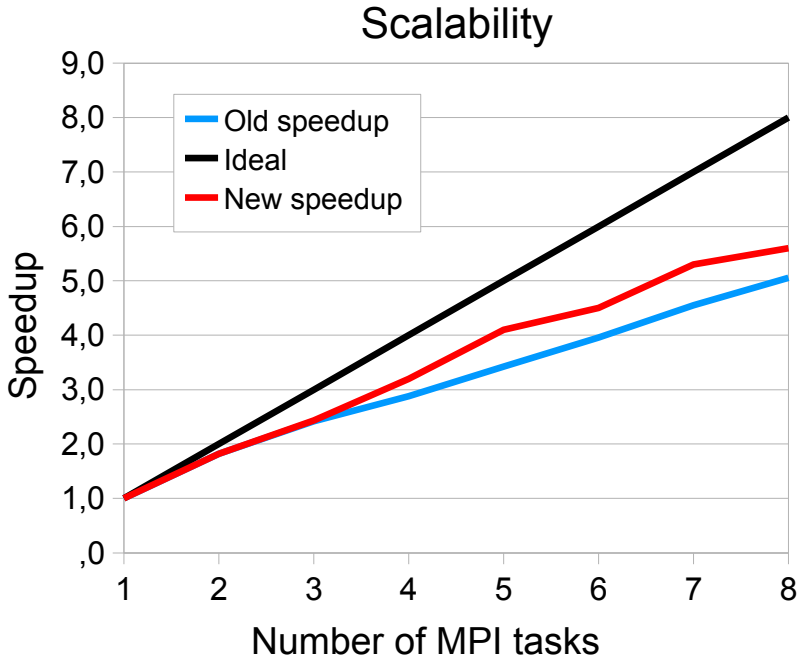
Avg-(%)	Avg.time (s)	Min.time (s)	Max.time (s)	St.dev (s)	Imbalance (%)	Calls (#)	Name
19.66	28.84	26.38	31.29	3.47	15.68	25176	CASGRA
19.48	28.58	26.76	30.40	2.58	11.98	25176	CASGQA
16.72	24.52	22.92	26.12	2.26	12.23	37827	CASPIA
3.76	5.51	0.00	11.02	7.79	100.00	56	ODBMP
3.67	5.38	0.00	10.76	7.61	100.00	20	LFIFER_MT

- Less load imbalance & less calls
- CASGQA still expensive but now in position #2
- Still a lot of trigonometric computations (CASGRA)
- CASPIA is raising

CANARI on scalar platforms

Optimized Canari *(Canari-Arome-France : 750 x 720 gridpoints, 60 levels)*

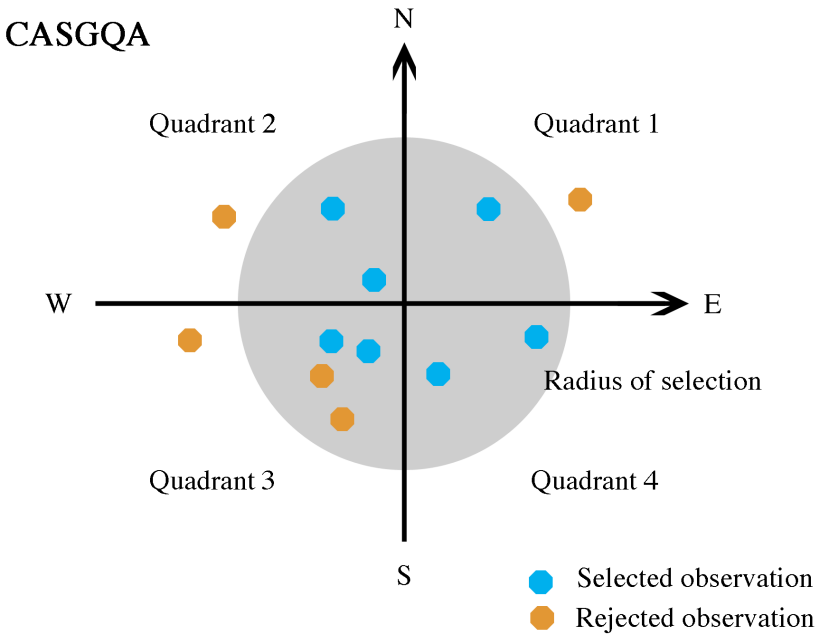


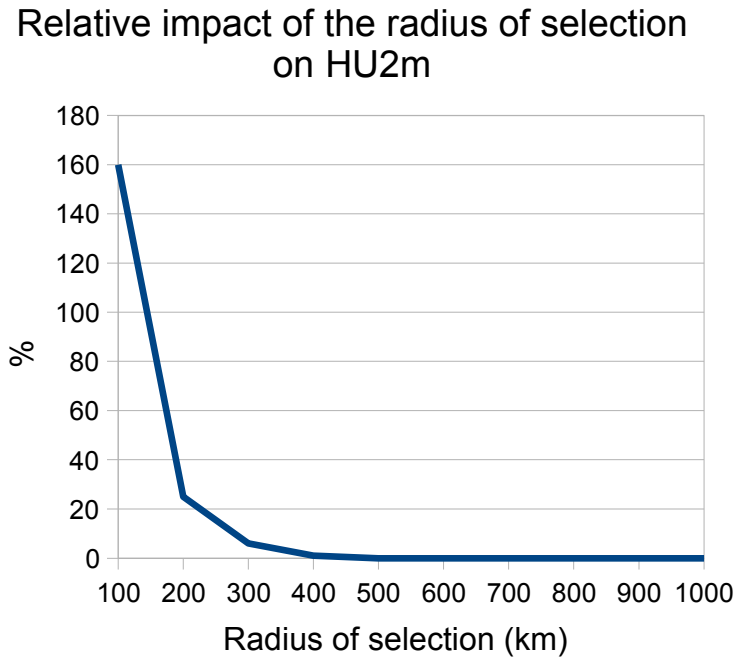


- **Real time saved after optimizations : ≈ 42 %**
- **Scalability : better**
- **Memory cost : reasonable** : 1 Gb (1 proc) to 3 Gb (8 procs)

CANARI on scalar platforms

Back to CASGQA





If we dissociate the SYNOP and the SHIP in the algorithm, we could reduce the radius of selection for SYNOP from 1000 km to ≈ 500 km
=> CASGQA, CASGRA & CASPIA would be cheaper

CANARI on scalar platforms

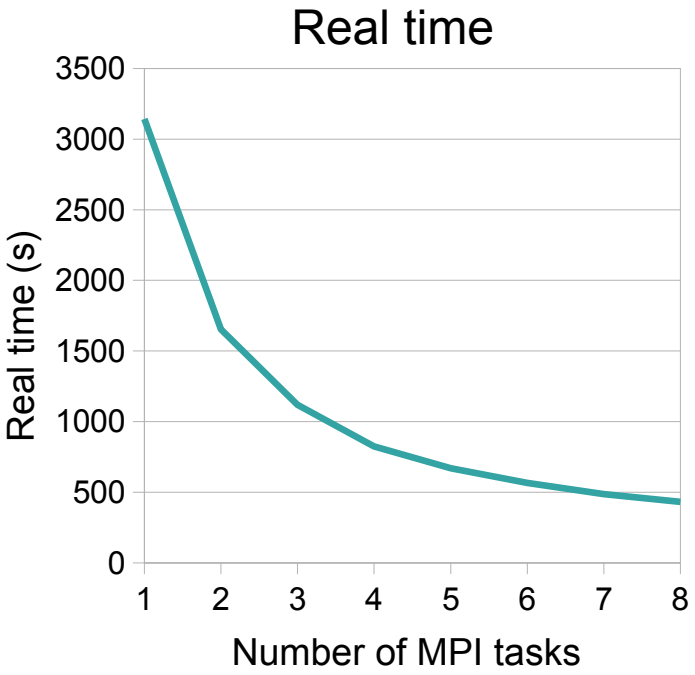
How the profile looks like when the radius of selection is set to 500 km (for 2 MPI tasks) :

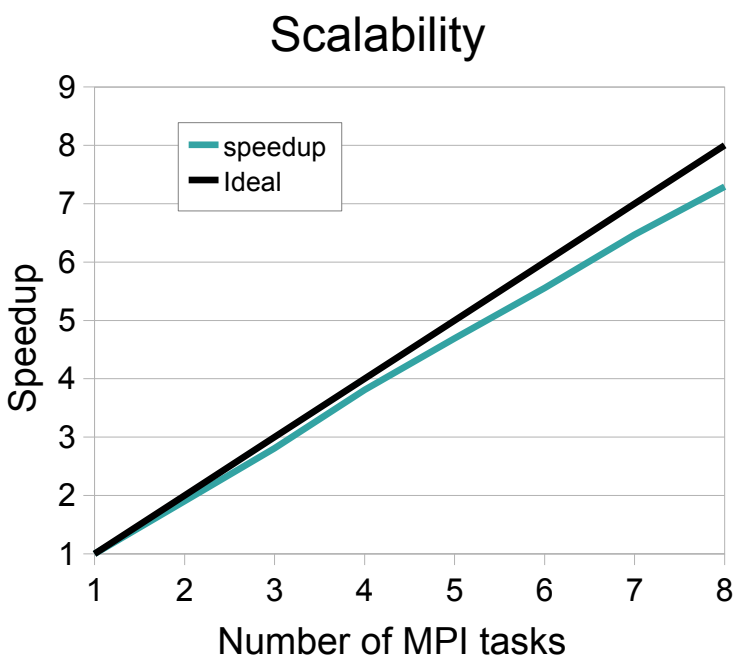
Avg-(%)	Avg.time (s)	Min.time (s)	Max.time (s)	St.dev (s)	Imbalance (%)	Calls (#)	Name
13.92	2.26	1.90	2.62	0.51	27.62	25176	CASGRA
5.79	0.94	0.00	1.88	1.33	100.00	56	ODBMP
5.46	0.89	0.00	1.77	1.25	100.00	20	LFIFER_MT
3.59	0.58	0.15	1.02	0.62	85.81	44	TRGTOL
4.65	0.76	0.76	0.76	0.00	0.00	2	SUESTAONL

- Real time : 18 s. instead of 154 s.
- CASGQA & CASPIA for free

CANARI on vector(1) platforms

(Canari-Arome-France : 750 x 720 gridpoints, 60 levels)





- **Scalability : good**
- **Time cost : much bigger than on scalar platforms**

(1) NEC SX9

CANARI on vector platform

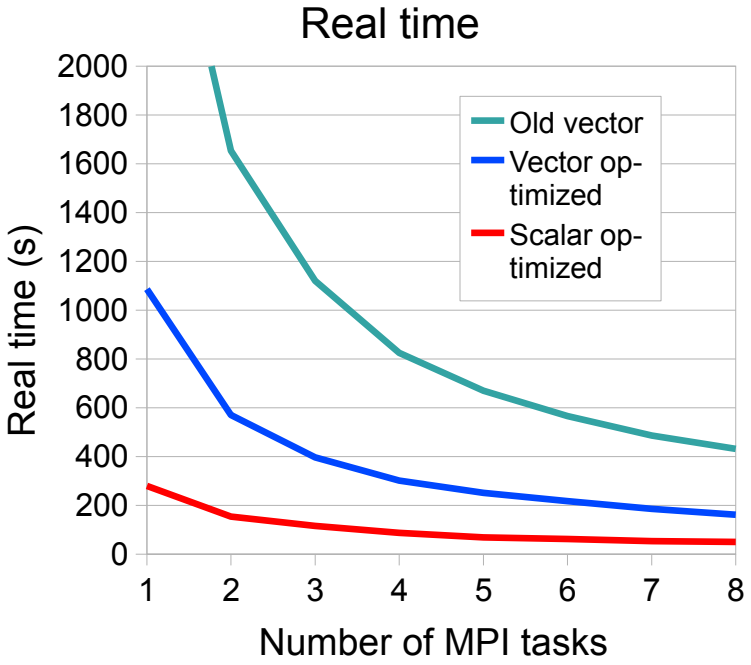
Performance is penalized mostly by **caspia.F90**

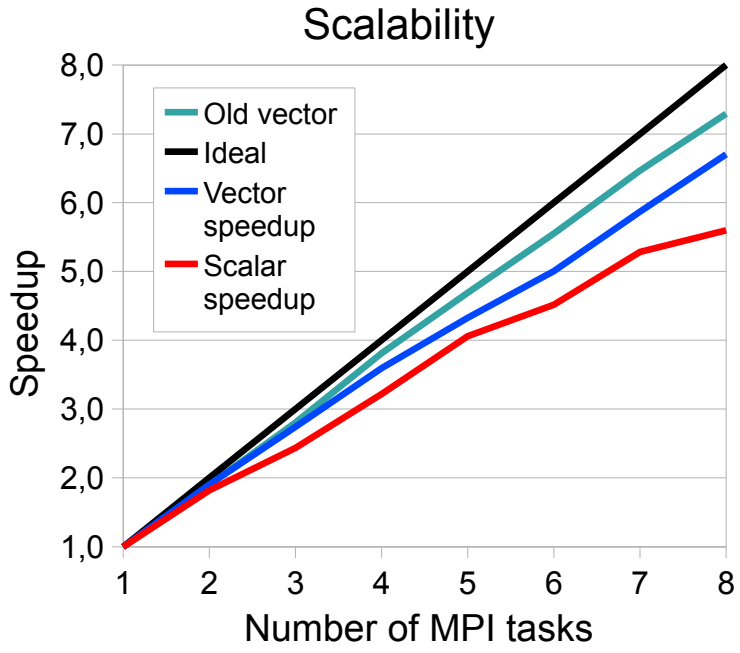
Unfortunately, the vector length
cannot be driven by the user (NPROMA) :
it is bound to the number of observations for a given point

- caspia.F90 has been substantially re-written to increase its vectorization rate and its vector length, thanks to various techniques :
  - **Loop fusion (++)**
  - **Loop collapsing (+++)**
  - **Loop pushing (+)**
- Various other vectorizations, mainly in **canada.F90**

CANARI on vector platforms

Optimized Canari *(Canari-Arome-France : 750 x 720 gridpoints, 60 levels)*





- **Real time saved after vectorizations : ≈ 64 %**
- **Scalability : better than scalar platforms**
- **Still much slower than on scalar platforms**

Integration aspects

The surface analysis with Surfex is composed of 3 applications run successively :

- (Fullpos) : append missing climatological fields (e923) and replacement of Surfex Ts by the previous surface analysis of Ts
- Canari : PBL fields +SST analysis
- Oi_main : Update surface fields for Surfex

Problems :

(though Fullpos and Oi_main are short tasks)

=> I/Os + extra-setup + scripts in between

=> Lack of multiprocessing support out of Canari

=> No ODB support in Oi_main

Integration aspects

Now possible : If it fully uses FA file format,
Oi_main can be called inside Canari
(or remain standalone in LFI)

To do : simplify fields setup in Canari (and in 3DVar as well)

Remaining problem :

- Oi_main is not a 1D algorithm => MPI works but reproducibility is broken

=> Workaround to do : provide a « global » SST field (F. Taillefer)

=> Alternative (as long as canari/oi_main are not too expensive) : *finish the open-mp support in Canari*

Conclusion

On scalar platforms,

- Canari runs now ≈ **1.7 times faster** than before
- Its **memory cost dropped** considerably

On vector platfoms,

- Canari runs now **2-3 times faster** than before
- However it is shaped for scalar machines, unless used for 3D OI

On any platform,

- **Separation of SYNOP and SHIP would make Canari even much faster**
- Integration of the side-applications will reduce the cost of the non-scalable parts

All the source-code modifications available in cycle 38T1

METEO FRANCE

Toujours un temps d'avance