Scalability of BATOR :

A problem of strong scalability ?

Ryad El Khatib (CNRM/GMAP)

Aladin Workshop / Hirlam All Staff Meeting   Norrköping, 05-08 April 2011

METEO FRANCE
Toujours un temps d'avance

Plan

INTRODUCTION

- Presentation of BATOR software
- Characteristics of BATOR compared to AROME

STUDIES

- Improvements and limits of the scalability of today
- Software performance
- Other parallelisations algorithms

CONCLUSIONS

- Recommendations for Bator and softwares in general
- Perspectives for Bator in particular

INTRODUCTION : présentation of BATOR

- Application to transform the collected observations over the planet into a database of the « ODB » format, suiatable for ARPEGE, ALADIN, AROME

- First task on the **critical path** of an assimilation suite 3DVar (AROME) or 4DVar (ARPEGE)

- Mechanism : several executions of the applications in order to transform sets of observations files delivered in different formats (BUFR mainly) and different sizes

« Anatomy » of BATOR as used for AROME (3DVAr)

Kind of observations or instrument	Number of files	Format	Size (Mb)
Surface	1	OBSOUL	1
Wind profilers + GPS	2	OBSOUL	1
Conventional	1	OBSOUL	7
SEVIRI	1	GRIB	18
HIRS	1	BUFR	2
AMSUA	1	BUFR	1
AMSUB	1	BUFR	4
SSMI	1	BUFR	3
IASI	1	BUFR	13
Geowind	1	BUFR	2
ERS + ASCAT	2	BUFR	1
AIRS	1	BUFR	0
RADAR	24	BUFR	200

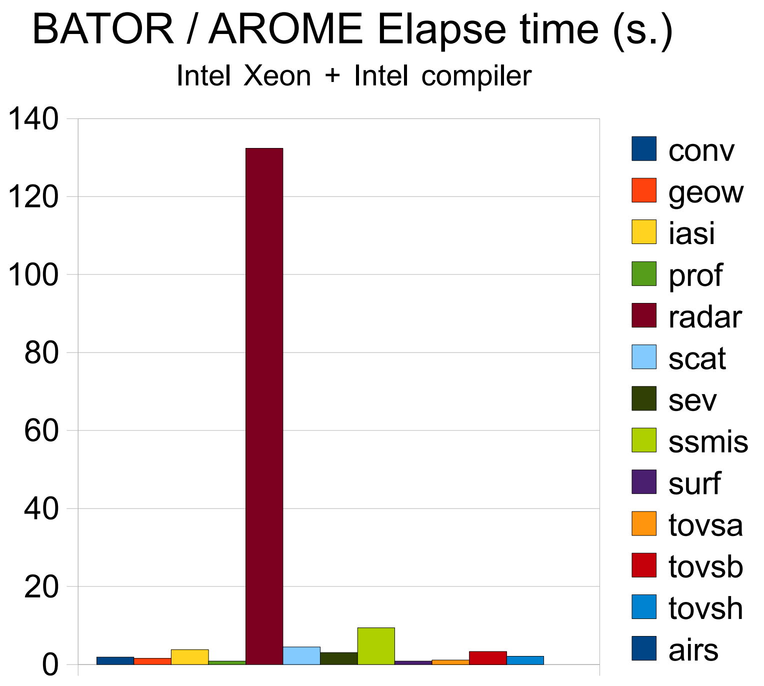
Characteristics of BATOR

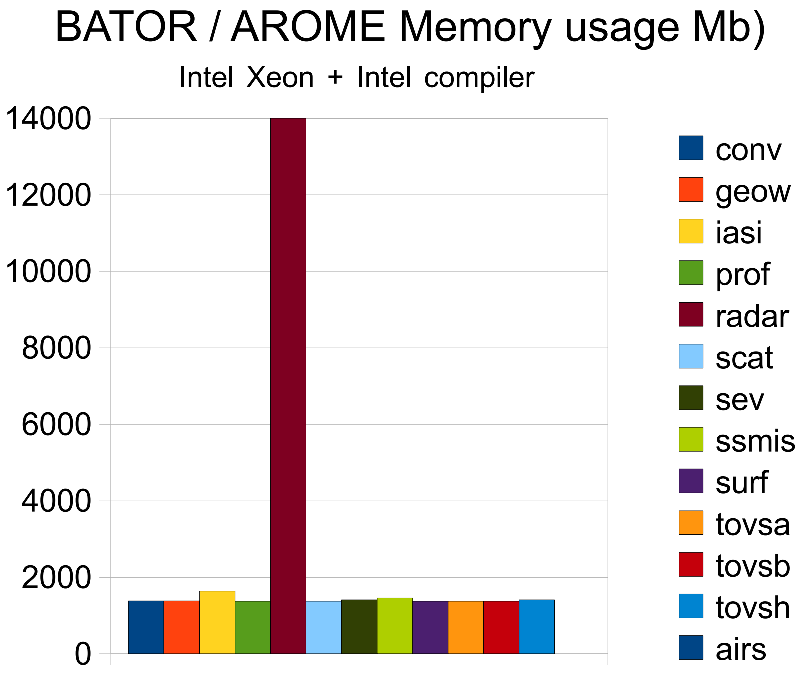
BATOR vs AROME	BATOR (without ODB)	AROME forecast (3h)
Number of lines of code	≈ 7 000	≈ 1 600 000
MPI parallelisation	Oui mais inefficace	Oui
Open-MP parallélisation	Non	Oui
CPUs used in operations	1	16 (SX9)
Elapse time	≈ 500 s.	≈ 500 s.
Memory per CPU	15 Go	11.5 Go (SX9)
Static memory allocated	≈ 600 Mo	≈ 400 Mo

Impact of the hardware architecture	NEC SX9 Vector machine	Intel Xeon Scalar machine
Elapse time	≈ 500 s.	≈ 180 s.

Load balancing of the BATOR tasks

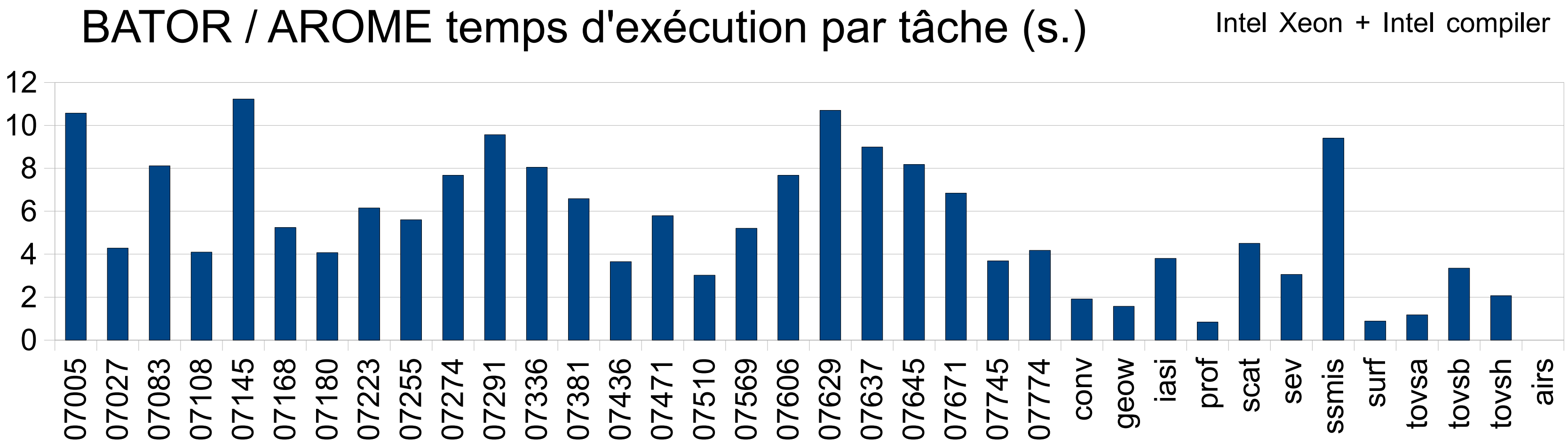
The task devoted to the 24 radars files is dramatically proeminent

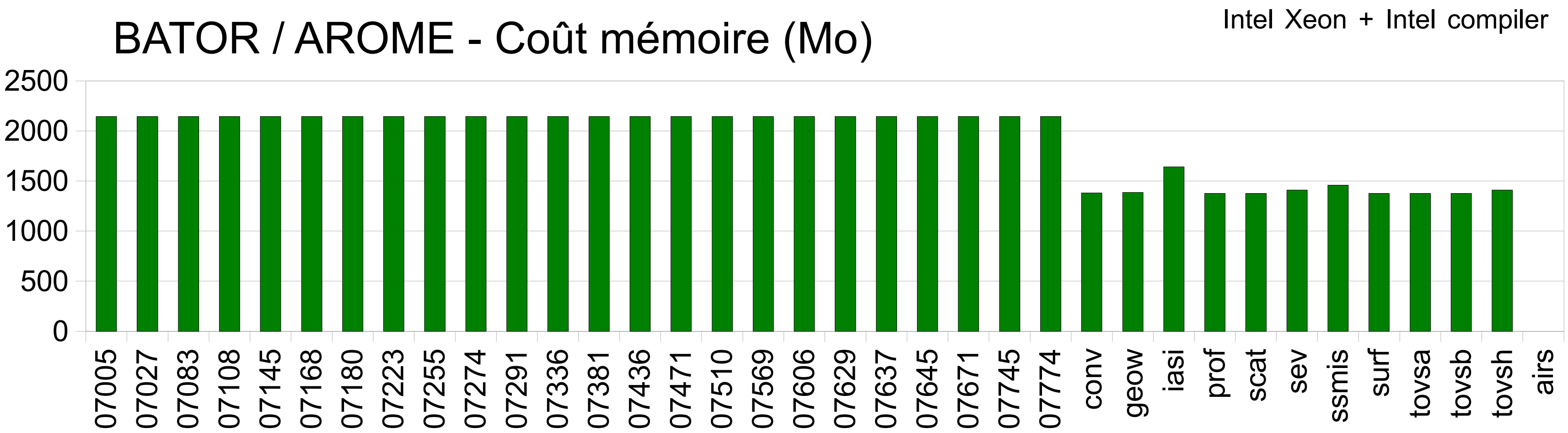




=> In such conditions, the scalability is near to zero

What if we maximize the number of tasks ?
(1 task per file)





=> Will a external dynamic load balancing be enough ?

METEO FRANCE
Toujours un temps d'avance

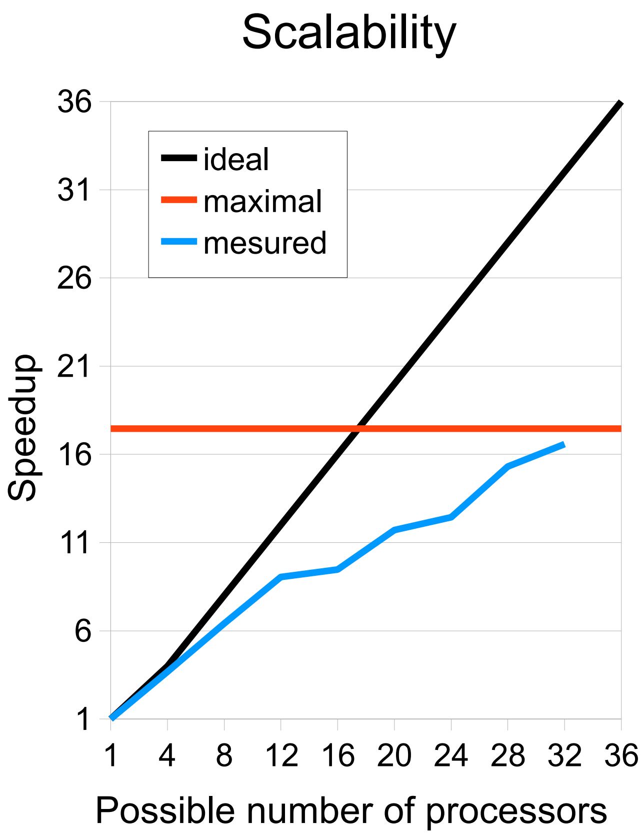
Limits of the scalability with a dynamic load balancing

- **Limited because the number of observations files is limited** ('36' wall)
- **Scalability loss because of residual load imbalance**

  (we can't run faster than the slowest task : red line)

And also :

- **Relatively high memory cost per task**
- **Memory-anti-scalable parallelisation scheme**



Practically : beyond16 procesors, the ressources at disposal is critical

METEO FRANCE
Toujours un temps d'avance

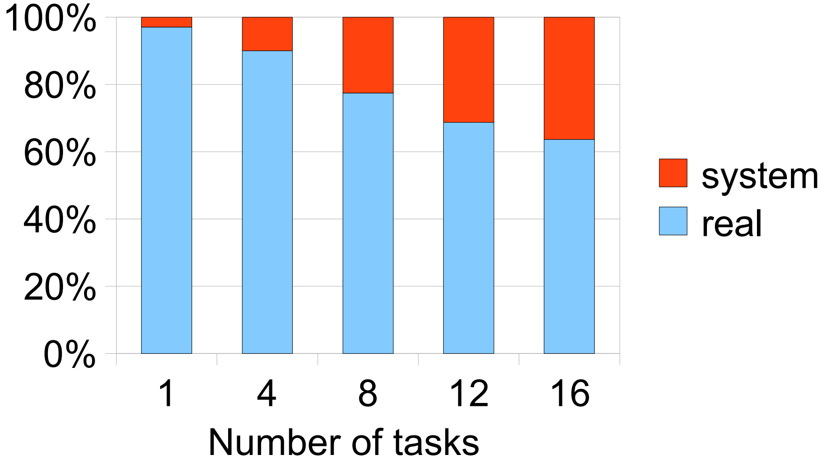
How to cross this scalability barrier ?

Jump over the obstacle ?

- Increase the number of observations files ??
  - In 4DVar : slice the files into shorter time slots
  - Cut the files per geographical sub-area ?
  - Define a better-adapted file format ?

  - *However, handling many small files may not be the best solution*

System time vs / Real time



METEO FRANCE
Toujours un temps d'avance

How to cross this scalability barrier ?

Turn around the obstacle, looking for better performance ?

- **Efficiency** may contribute to improve the **Scalability**
  - Are the files read/written efficiently ?
  - Does the algorithm fit parallel machines ?
  - Is the code performant ?

```
 Avg-%   Avg.time # of calls : routine
42.65%      9.697      13312 : BUEXS3
26.40%      6.002          7 : Bator_lbufr_radar
 7.89%      1.795          1 : BATOR
 4.87%      1.107          1 : BATOR_ELIM
```

=> An obvious efficiency issue in decoding radar BUFR files

=> Subroutine Bator_lbufr_radar to be further examined

How to cross this scalability barrier ?

Turn around the obstacle

Using other directions in parallelism ?

- BUFR decoding library uses **global variables**
  - => To use **Open-MP** one should modify the software
- Bator algorithm is ***intrinsincally* sequential**
  - => To use **Open-MP** one should revisit the algorithm
- Bator contains a lot of **loops left by GOTO instructions**
  - =>Difficult to analyse the code performance and implement **Open-MP**. The code has to be modified.
- **MPI parallelisation** in dans Bator : it exists but :
  - Parallelism based on the distribution of a set of input observations files
  - => No treatment of **memory load balancing**
  - => No treatment of **CPU load balancing**
  - => finally less efficient than the external dynamic parallelisation

Another unexpected issue

The number of observations pools should be a multiple of the number of MPI tasks in the subsequent applications

(Screening, Minimization)

- ODB_IO_METHOD=1
  - 1 file per table and per pool
  - => would lead to much small files on many-processors machine. Is the file system ready to support this ?
- ODB_IO_METHOD=4
  - Less files of fixed size
  - => Requires (much) more memory. May easily break the memory limit of a node with Bator on a scalar machine
- Alternative # 1 : ODB_IO_METHOD=1 + tool « Odb1to4 »
- Alternative # 2 : ODB_IO_METHOD=4 + « reshuffling » (needs a specific ODB recompilation)

Conclusions

- Bator exhibits strong scalabilitiy issues than, could be overcome :
  - Better **I/O conditionning** (format, number of files)
  - **Parallelisation methods** (MPI, threads) **using algorithms adapted to the problem**
  - Playing with ODB tools

- The search for **scalability** should not mask the **performance issue**

- Softwares should **evolve** permanently according to its **context of execution**, not its own being :
  - « High Performance Computation » => *batch processing* (« vectorization »)
  - Evolution of programing languages, hardware architectures
  - Software context (3DVar, 4DVar for Bator, OOPS later on)

Perspectives for Bator

- Scalability and performance issues for Bator/AROME could be solved for short or mid term :
  - Thanks to a sufficient external parallelisation
  - Because the enhancement of performance (Bufrdc) seems feasible

- Bator/AROME-3DVar solution is extensible to 4DVar

- Ongoing : Fusion of ECMWF Bufr2odb with Bator
  - Full parallelisation support from Bufr2Odb
  - Get the software out of the critical path thanks to an earlier upstream execution
  - Object-oriented context for 3DVar/4DVar ?

METEO FRANCE

Toujours un temps d'avance