ROOT
Status & Roadmap

Rene.Brun@cern.ch

26 March 2007

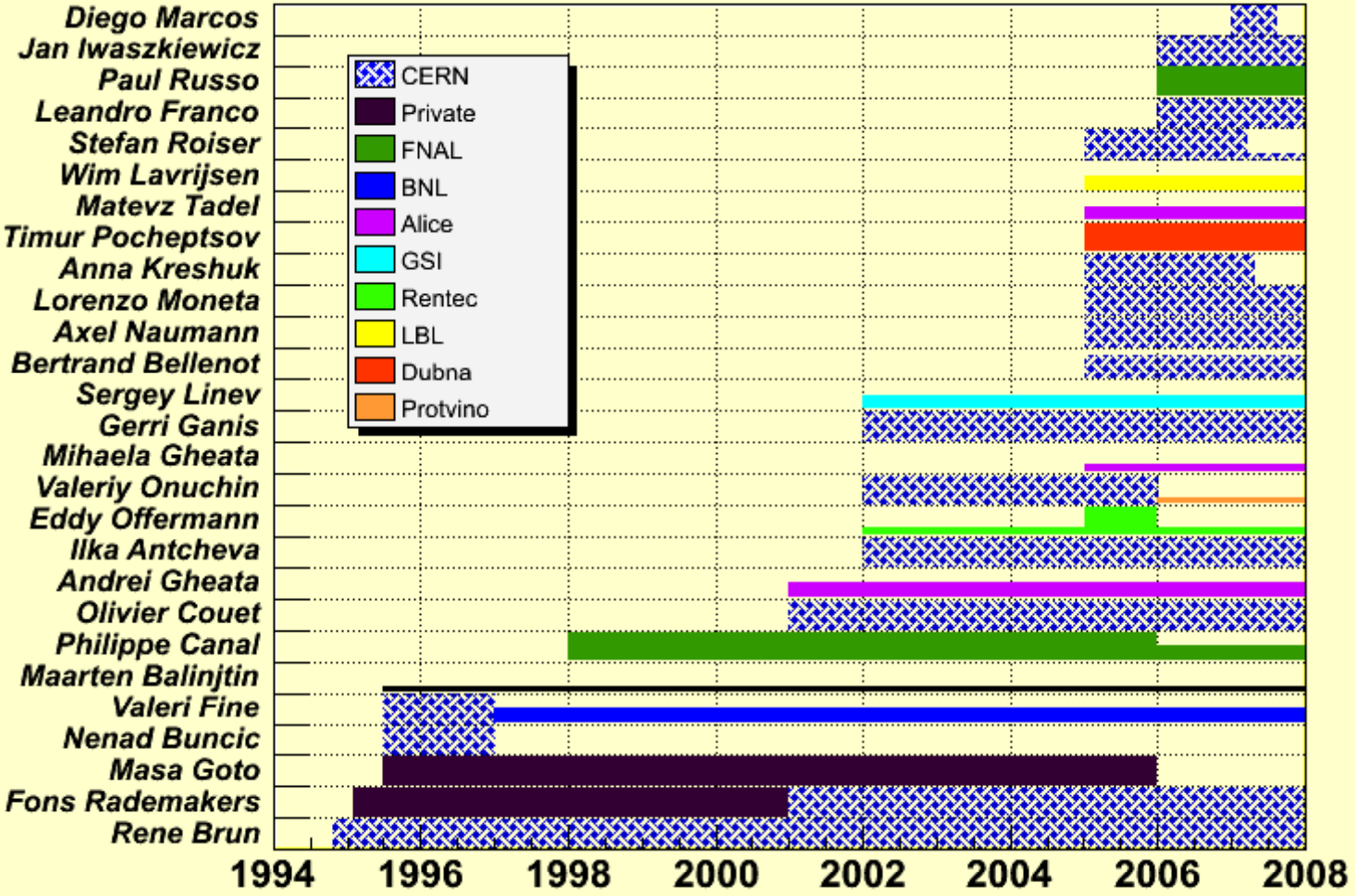
Plan

- Quick review of main features since the previous workshop (September 2005).
- More details later by the work packages
  - CORE: Fons
  - I/O: Philippe
  - CINT: Axel
  - Math: Lorenzo
  - Geometry: Andrei
  - Graphics: Olivier, Matevz
  - GUI: Ilka, Bertrand
  - PROOF: Gerri
- Current work (part of v5.16 in June).
- Road-Map: Next 18 months

Quelle histoire !!

- 1995: exile at Prevessin site
- 1996: CINT in ROOT
- 1998: FNAL, BNL choose ROOT
- 2000: Babar (Objectivity vs ROOT)
- 2001: Hoffmann review at CERN
- 2002: LCG starts. ROOT is a “provider”
- 2005: LCG phase2: becoming main stream

ROOT Team Evolution

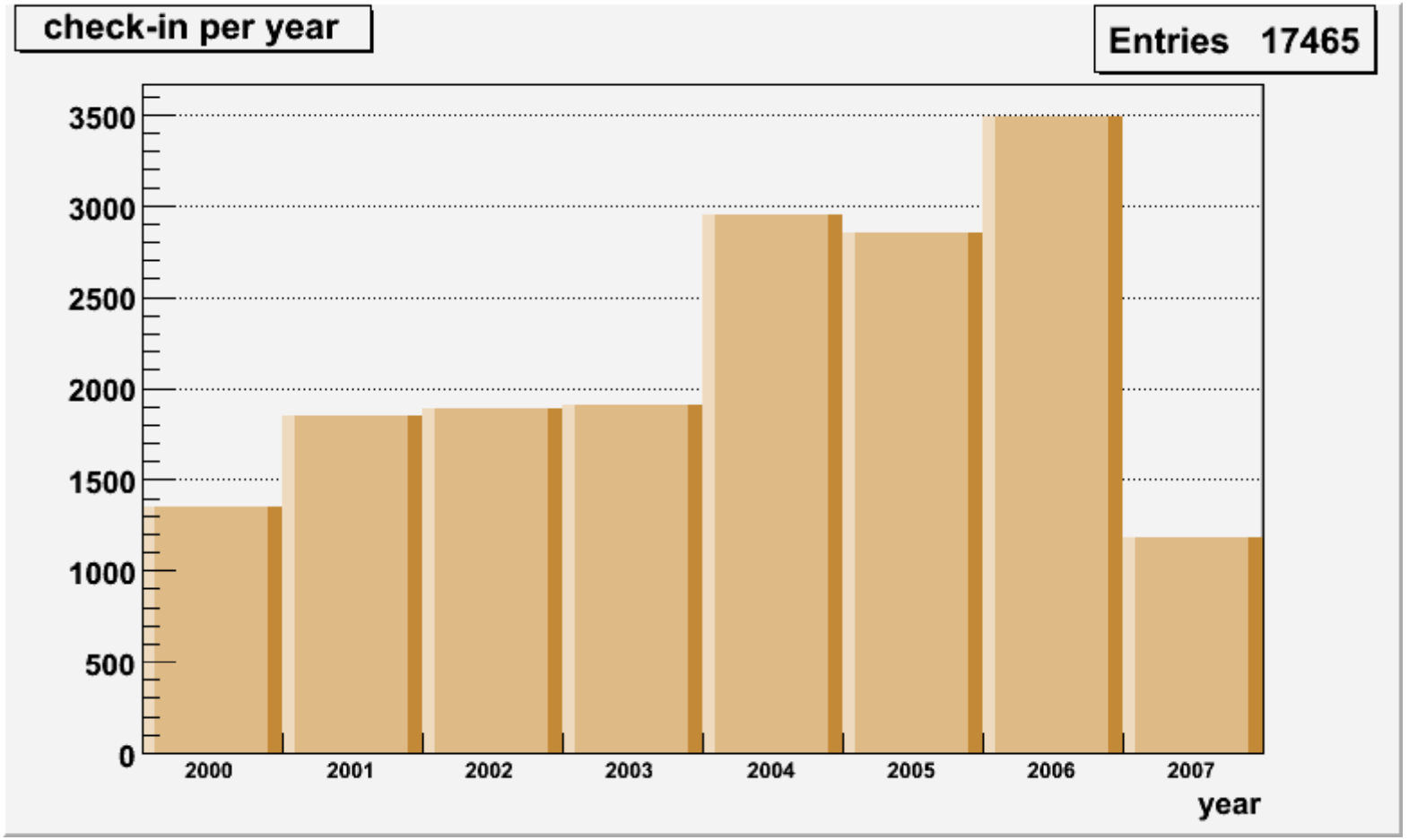


ROOT Team today
(working 50% or more)

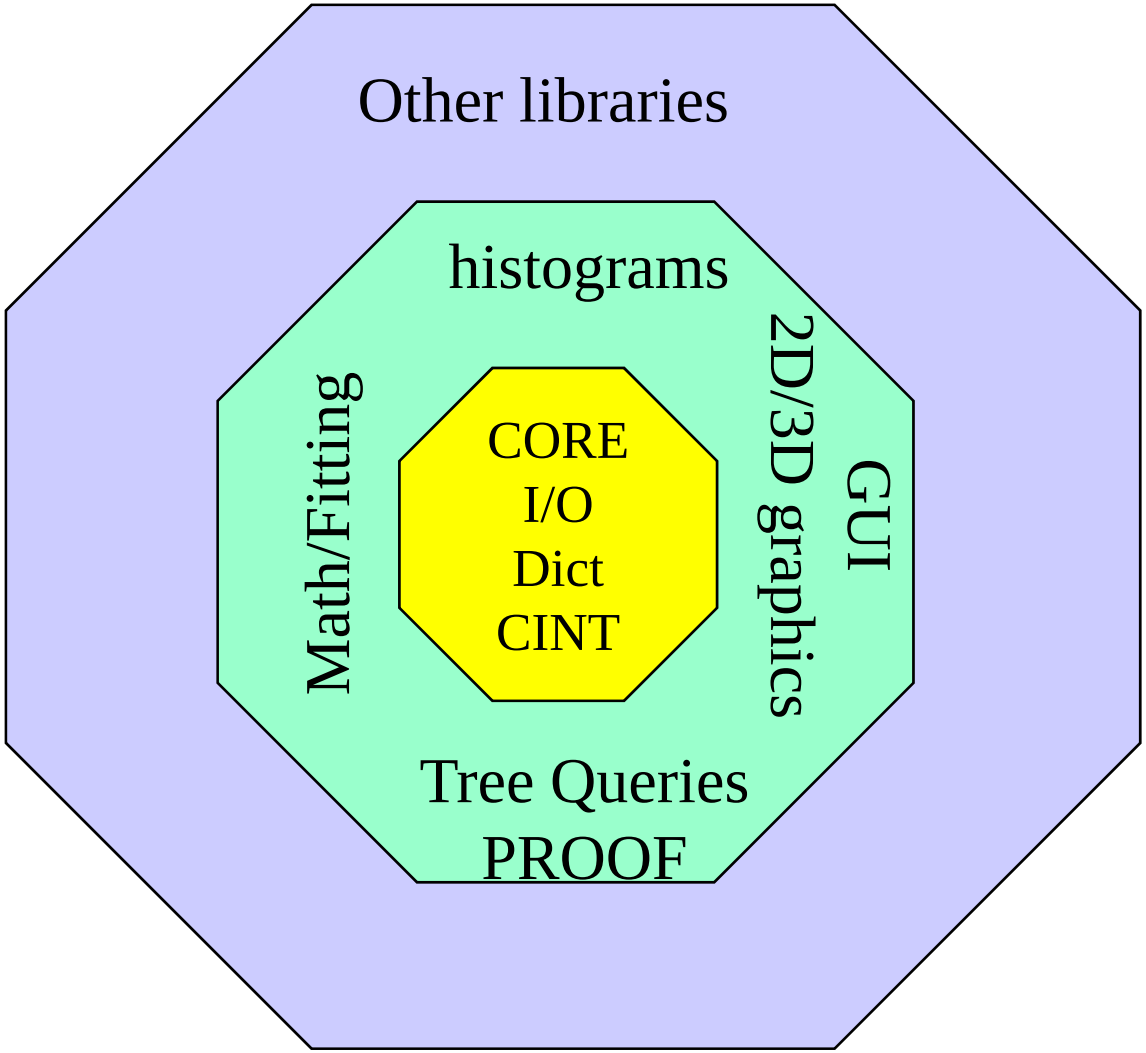
Interesting talks about all these topics

- CORE: Fons Rademakers, Leo Franco, Diego Marcos
- DICT: Axel Naumann, Philippe Canal, (Stefan Roiser)
- I/O: Philippe Canal(50%), Paul Russo
- MATH: Lorenzo Moneta, (Anna Kreshuk)
- GEOM: Andrei Gheata, Mihaela Gheata
- GUI: Ilka Antcheva, Bertrand Bellenot, (V.Onuchin(yy%))
- GRAF: Olivier Couet, Timur Potcheptsov, Matev Tadel(50%)
- PROOF: Fons, Gerri Ganis, Jan Iwaszkiewicz
- PYROOT: Wim Lavrijsen (xx%)

Code development activity



First Priority: Consolidation



First Priority: Consolidation

- It is vital for the 4 LHC experiments to have a stable and solid object persistency system.
- In the past two years, the main objective has been to implement the necessary features to support the object models from the experiments.
- We are now entering a crucial phase where stability and performance are the most important elements. Developments in the I/O area will be kept to a minimum with main focus on robustness and performance.

Prepare the Future

- After 12 years of intense developments, it is time to review the strengths and weaknesses of the system and correct the directions accordingly.
- We will continue to develop and improve the major sub-systems.
- New technologies are appearing
  - Hardware: multi-core cpus
  - Virtualization
  - GRIDs and various forms of parallelism
  - Automatic Updates
- ROOT is a large system used and installed by thousands of people. Simplifying the installation, the ease of use is becoming a must.

Use Patterns

- In fact, we see two conflicting requirements;
- Experiment frameworks: They need stable versions and are mainly concerned by the I/O, dictionaries and math sub-systems. In this case, the installation of ROOT is part of a complex experiment software distribution system.
- The end-users who want to install the latest greatest version on their laptops where the experiment frameworks are not always available.

Towards BOOT

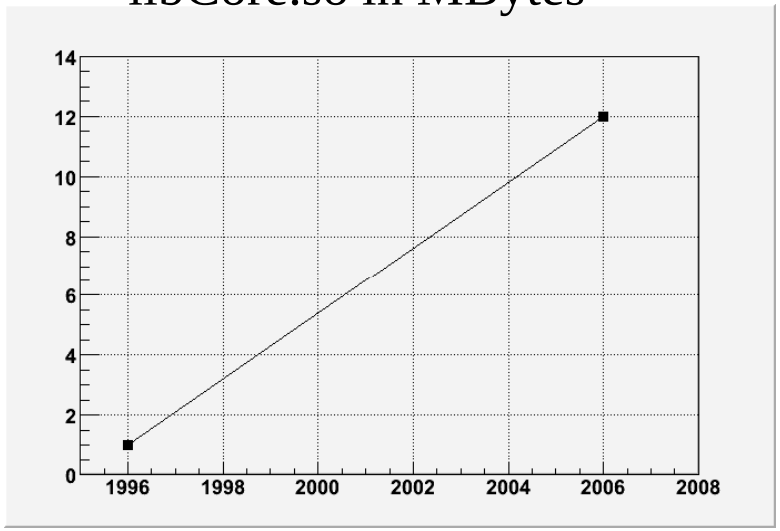
- At CHEP06 in Bombay the idea of BOOT was introduced. It will be described in more details in another talk.
- One of the most important ingredients for the success of BOOT is an optimum modularity and packaging of the system in small units dynamically loadable at run time.
- In the past few months, we have spent a lot of effort to rationalize the ROOT kernel to facilitate the future developments and the scalability.
- The next slides describe this work, including some recent and very important achievements.

More Modularity

- Improve modularity with new subdirs “io”, “math”, ”net”
- Reduce the size of libCore
- Reduce libraries dependencies
- Reduce compilation time when changing the most important header files.
- Better documentation in sources

Size of libCore

libCore.so in MBytes



5.10: Process Map (pmap) root.exe = 66 Mbytes

5.10: Process Map (pmap) root.exe -b = 58 MBytes

5.14: Process Map (pmap) root.exe = 44 MBytes

5.14: Process Map (pmap) root.exe -b = 34 Mbytes

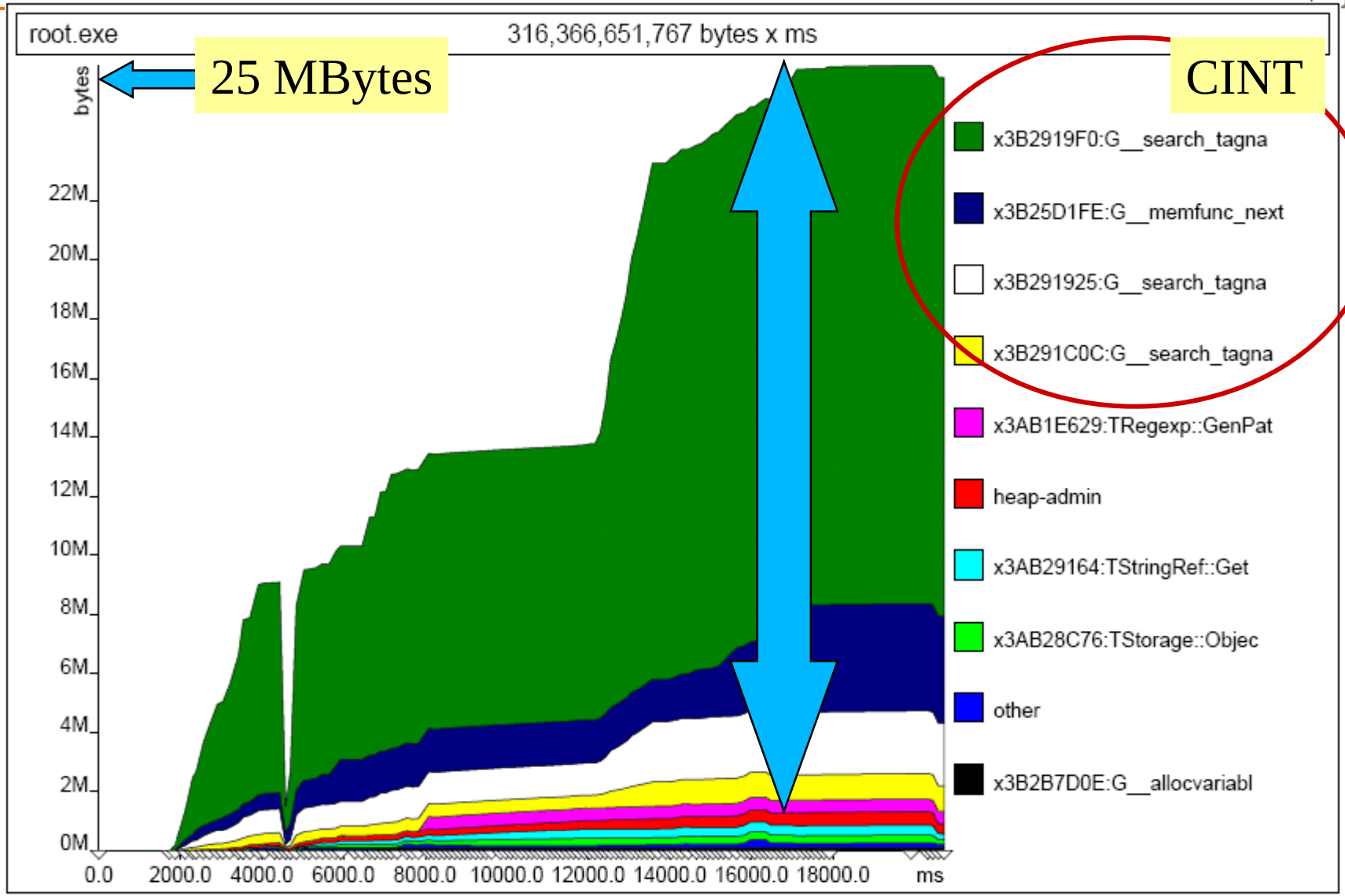
5.15: Process Map (pmap) root.exe -b = 17 Mbytes

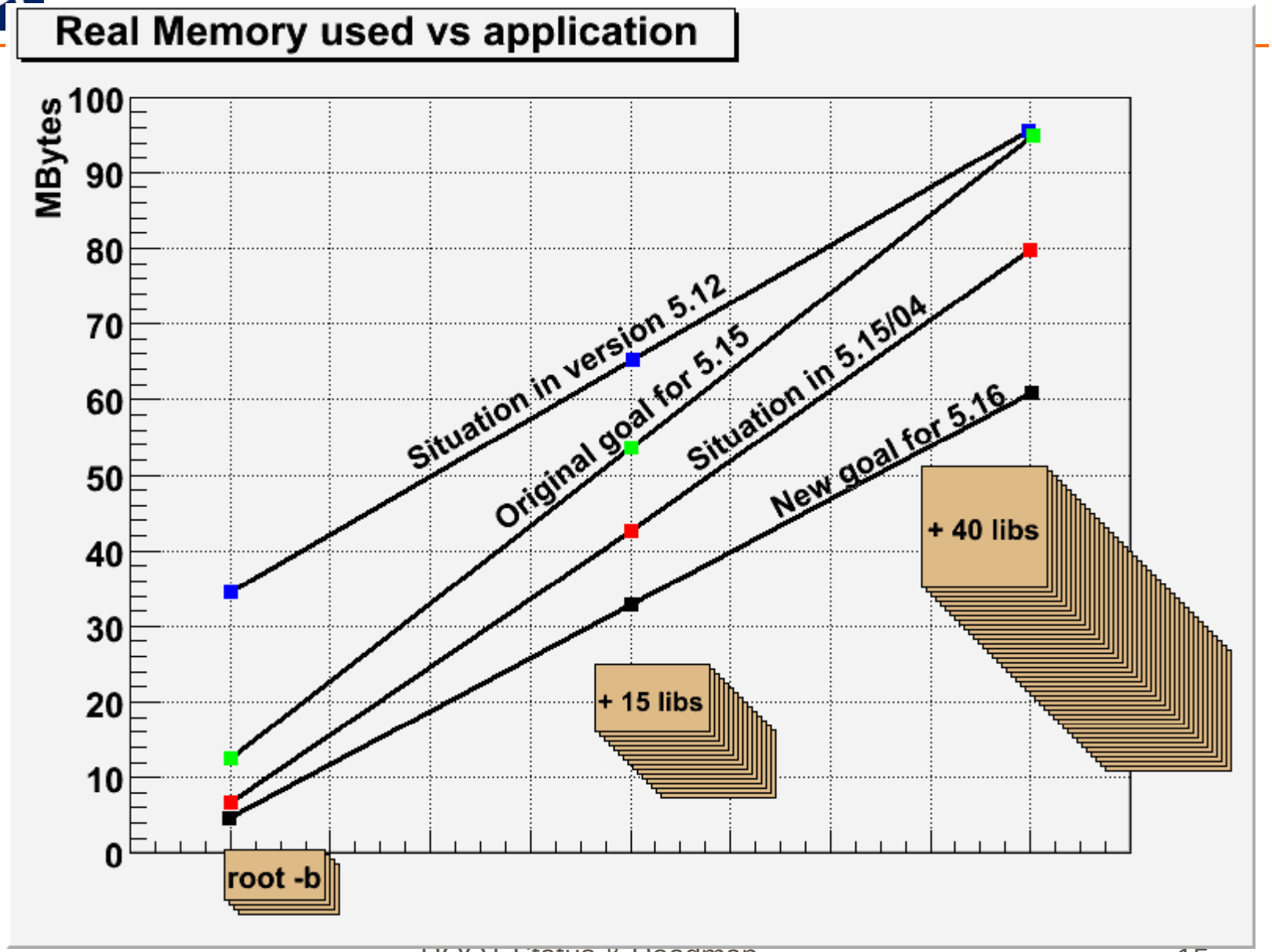
5.16 (June): Objective is to reach 10 Mbytes

Obtained by reducing size of CINT pointers tables

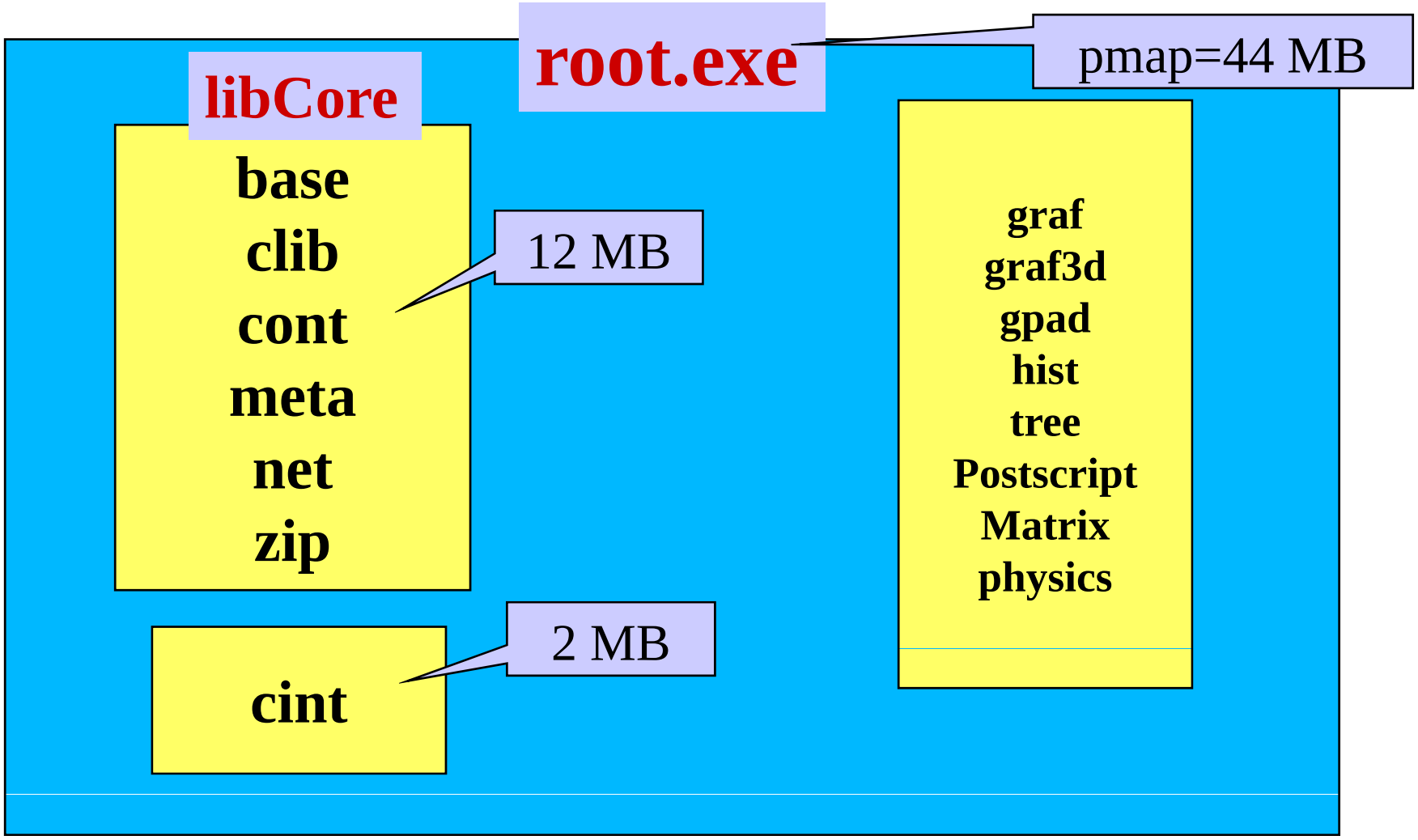
Reducing pmap reduces startup time

Heap size in 5.12
(as reported by valgrind massif)

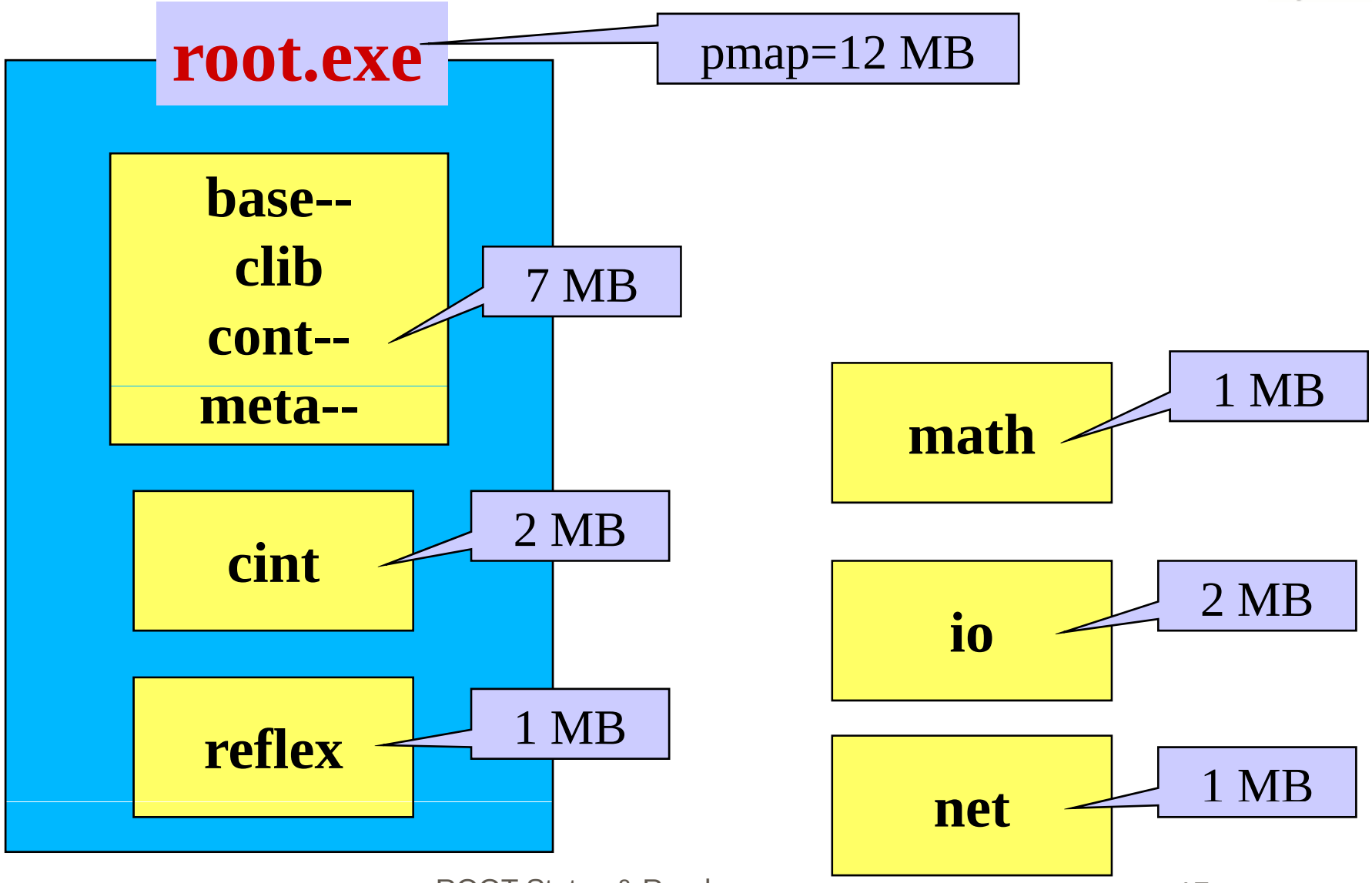



Real Memory used vs application
MBytes
100
90
80
70
60
50
40
30
20
10
0
Situation in version 5.12
Original goal for 5.15
Situation in 5.15/04
New goal for 5.16
+ 40 libs
+ 15 libs
root -b

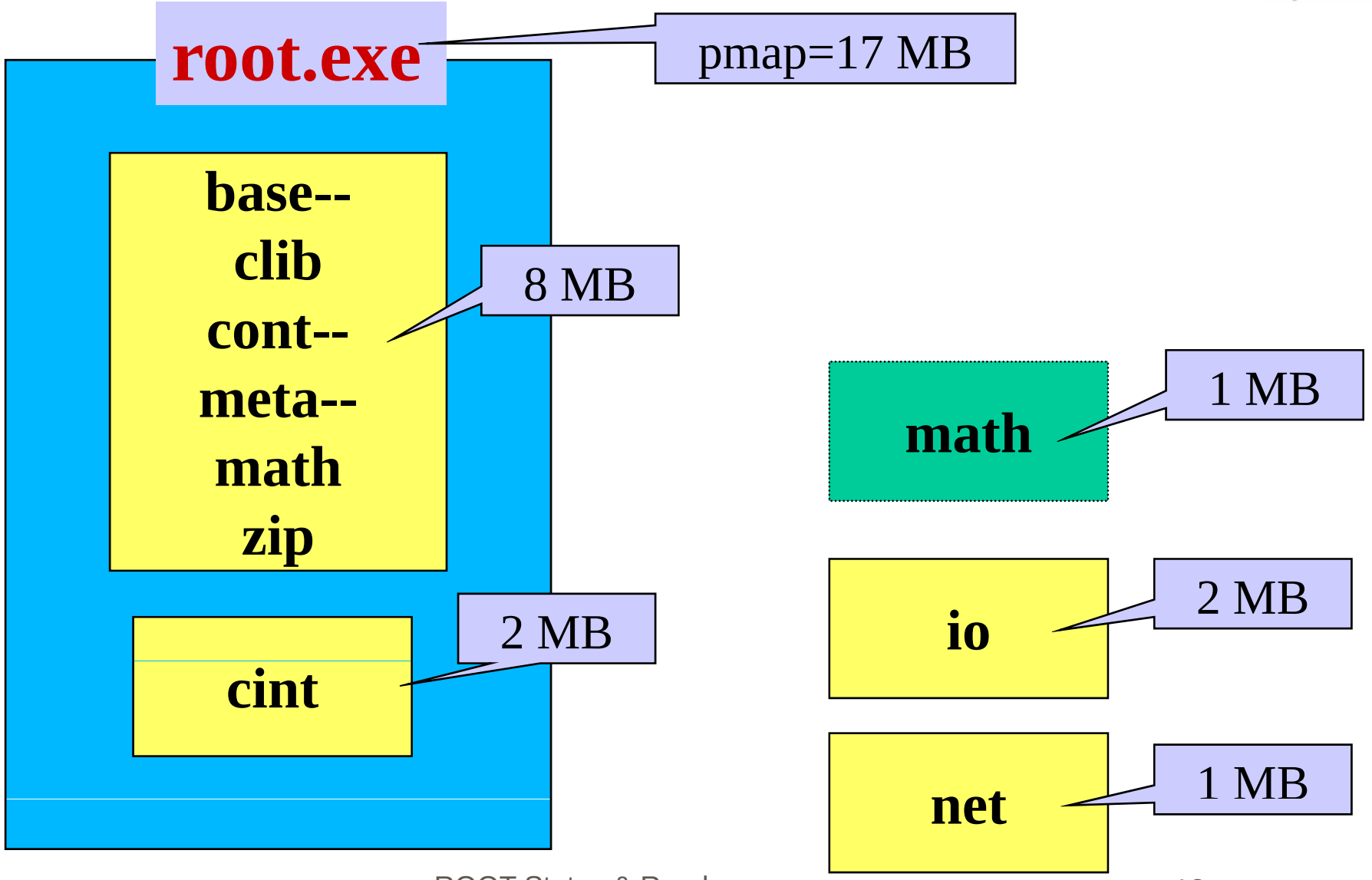
Libraries organization (5.14)



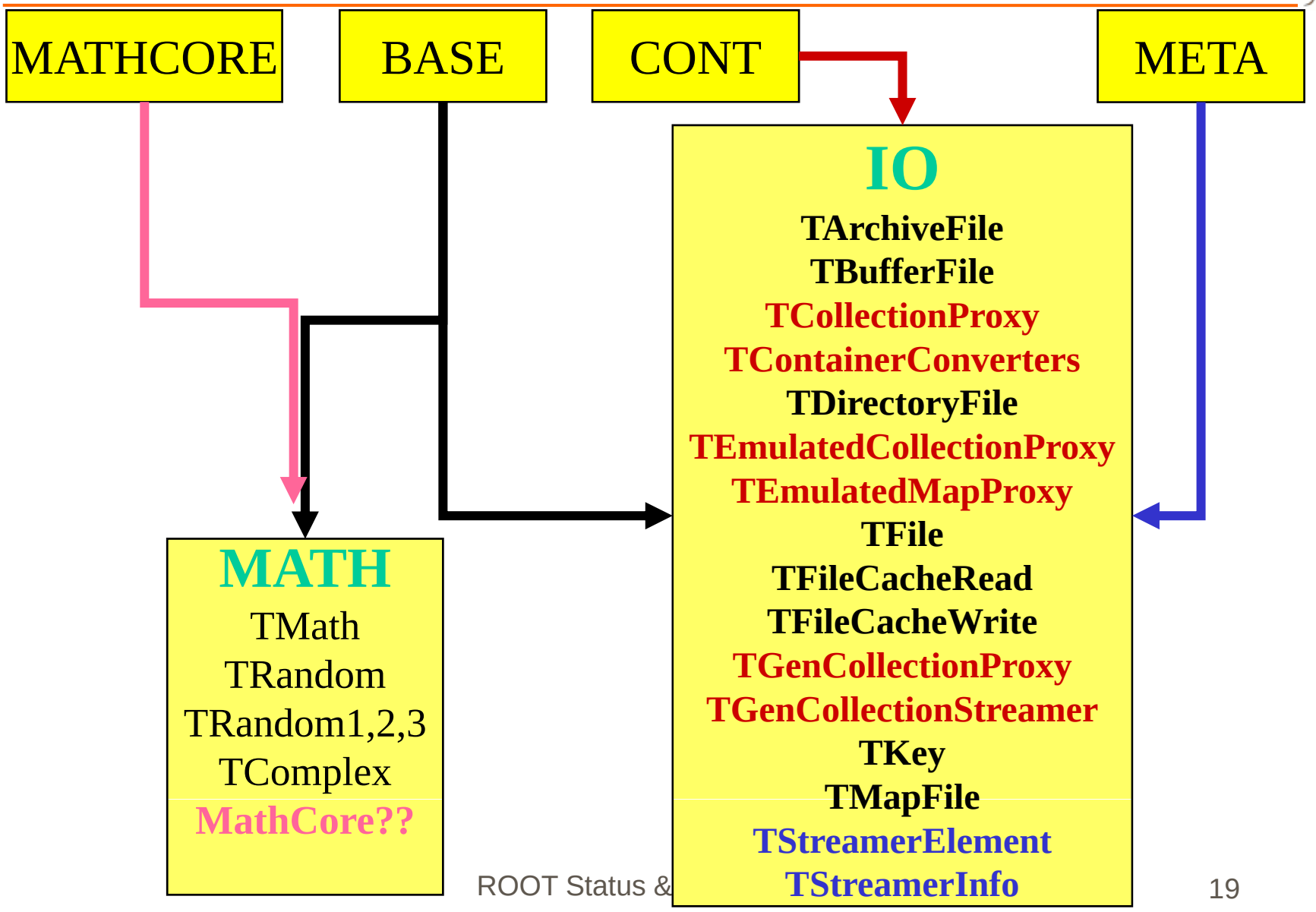
Libraries reorganization
(minimal goals for 5.16)



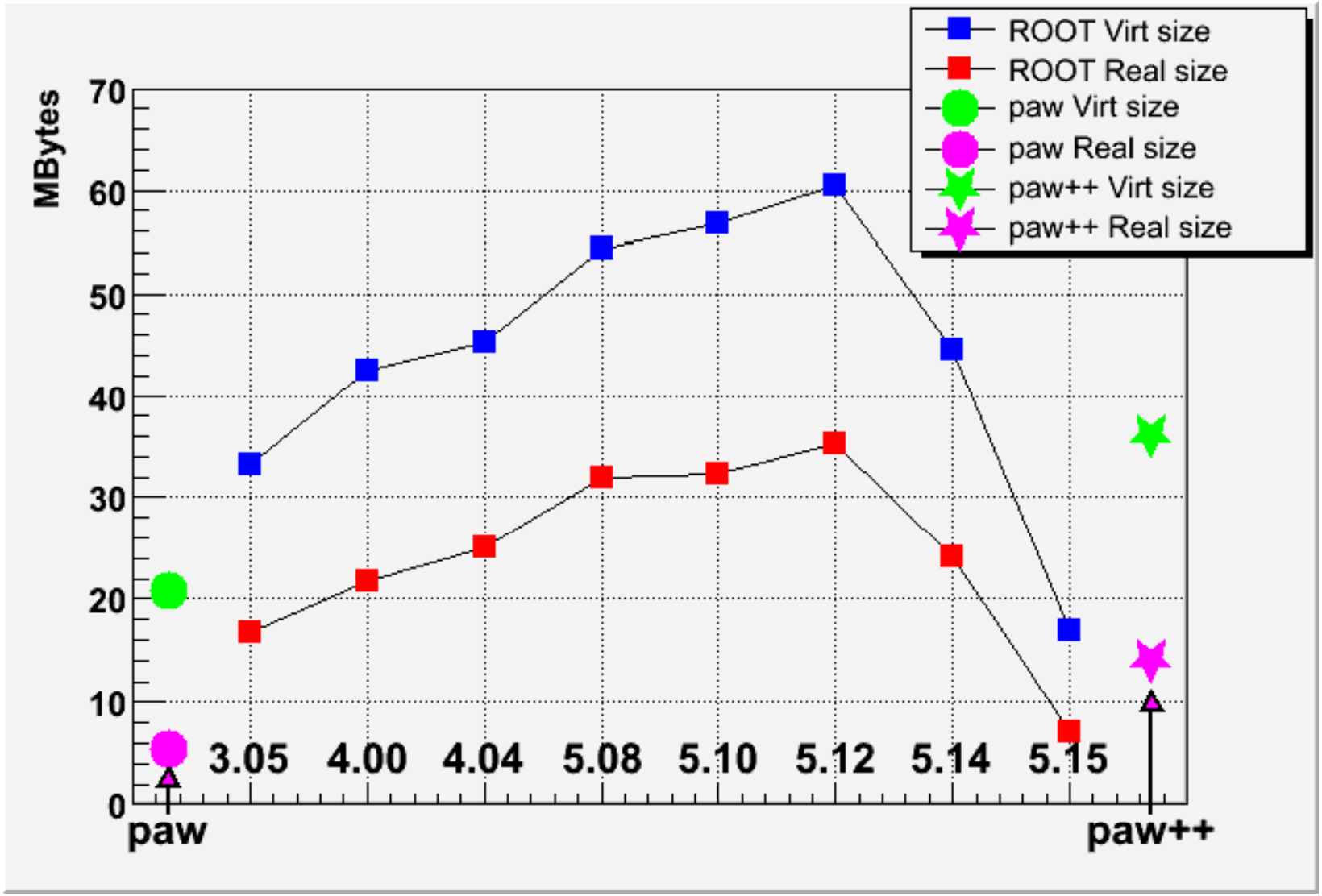
Libraries reorganization
(proposal phase1 today)

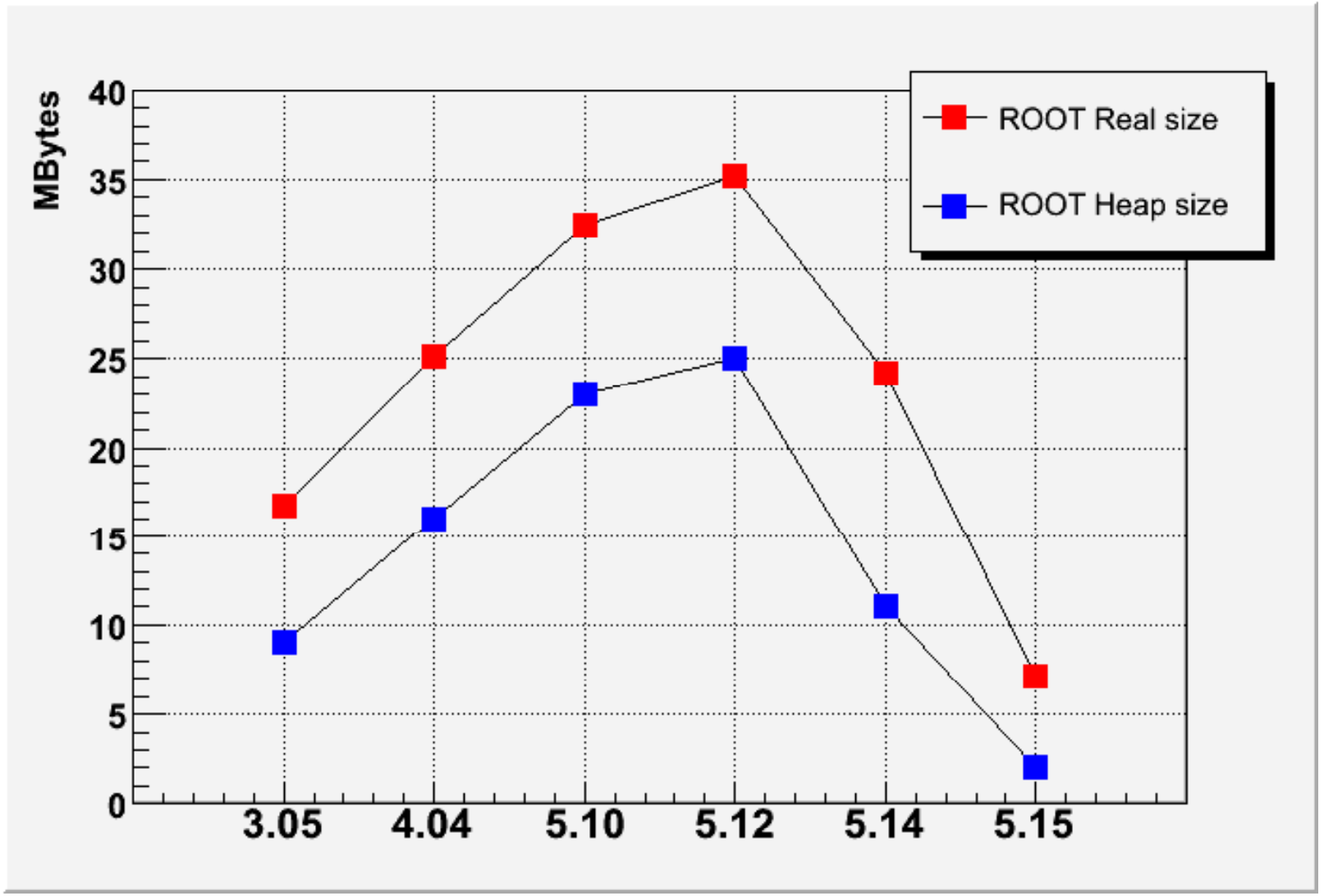


Cont / meta / base reorganization



Where are we today with 5.15/04 ?




MBytes
40
35
30
25
20
15
10
5
0
3.05
4.04
5.10
5.12
5.14
5.15
ROOT Real size
ROOT Heap size

Where are we today with 5.15/04 ?

	3.05	5.10	5.12	5.14	5.15
root -n	33.2 16.8	56.7 32.4	60.5 35.2	44.6 24.2	17 7.1
root	33.8 18.1	59.2 35.9	63.2 38.5	47.5 28	29 13.6
hpx.Draw	35.4 20.3	62.2 38.4	67.1 42.1	51.8 31.7	41.9 21
benchmarks.C	75.4 31.7	103.4 50	118.5 53.6	104.1 43.6	94.5 32

virtual size (MB)

real size (MB)

Important changes in CINT

- The data structures G__ifunc_table has been reorganized to be independent of the parameter G__MAXFUNCPARA = 40. Only real function arguments are stored in memory.
- The data structures G__inheritance has been reorganized to be independent of the parameter G__MAXBASE = 50. Only real function arguments are stored in memory.

Important changes in AutoLoading

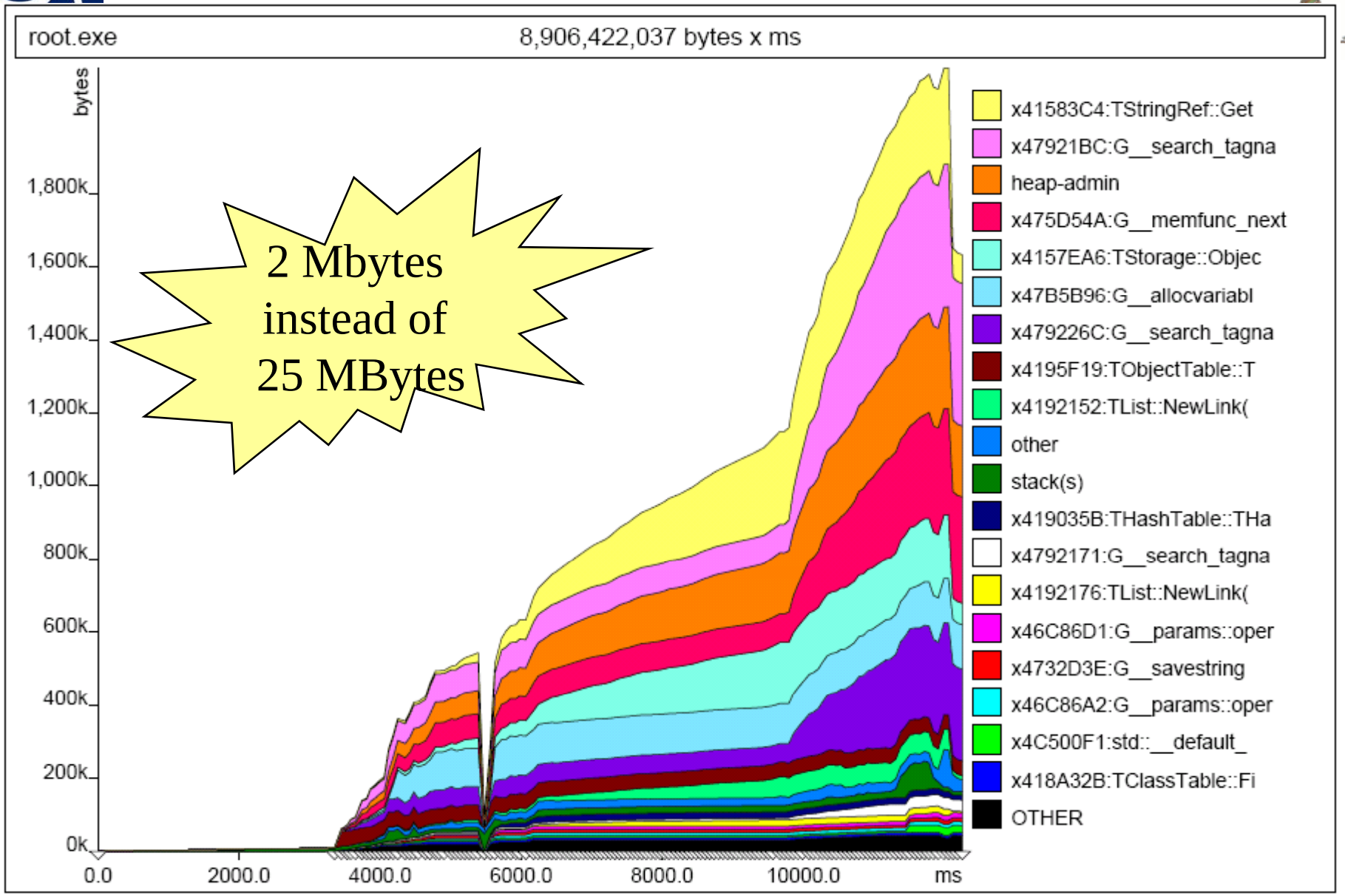
- The reorganization of the libraries has required several changes and improvements in the auto loading system (see Fons talk)

Effect on compilation time

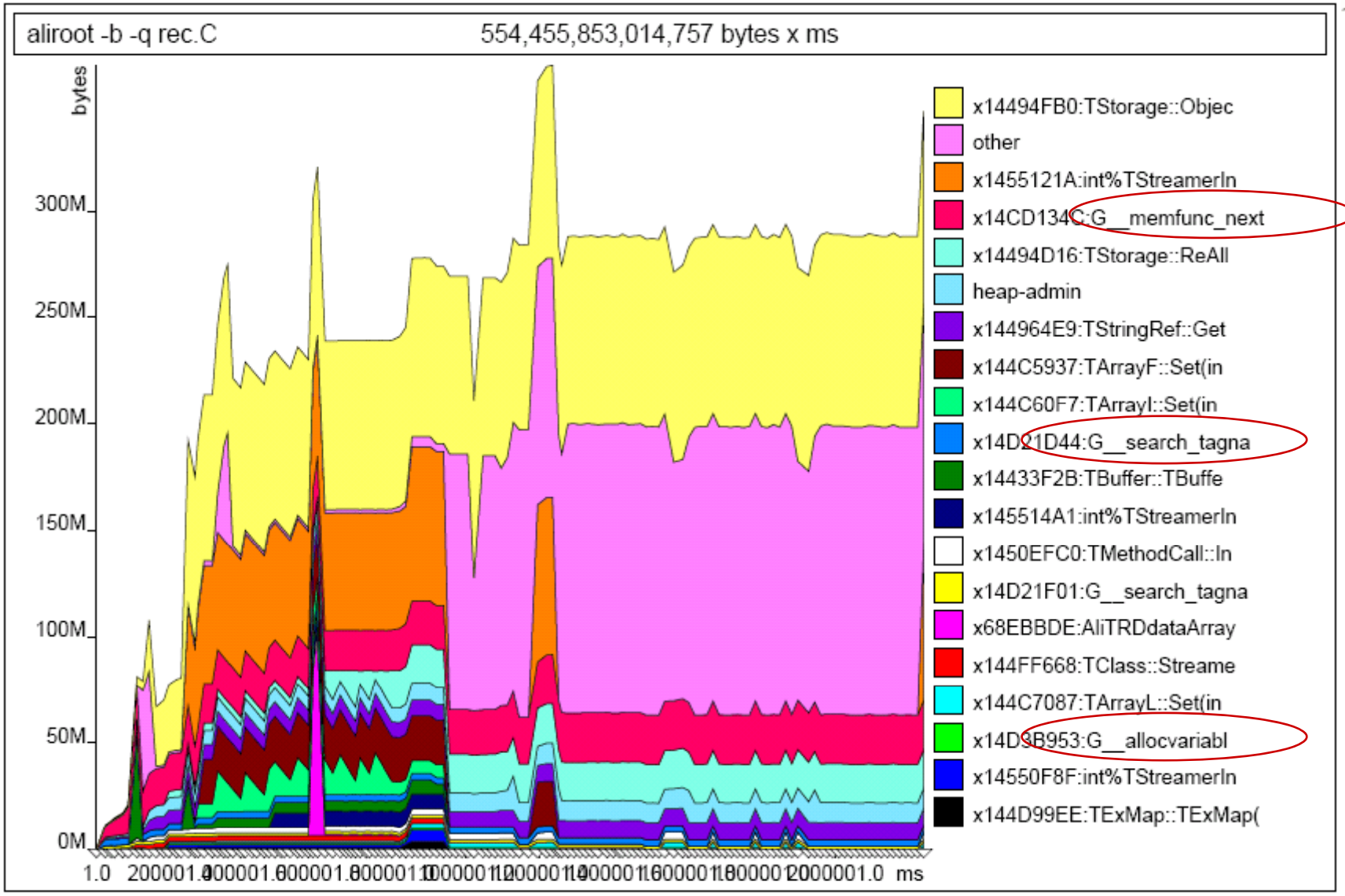
Time for “make –j 2” on MacBookPro when all packages enabled

	5.14	5.15 (CVS)
Full compilation	20’ 43”	20’ 12”
Varargs.h	15’ 31”	7’ 01”
TQObject.h		7’ 50”
TMath.h	14’ 37”	4’ 11’’
TBuffer.h	15’ 00”	14’ 46”
TBufferFile.h		1’ 44”
TDirectory.h	13’ 59”	5’ 38”
TDirectoryFile.h		1’ 57”
TROOT.h	13’ 54”	5’ 01”
TStreamerInfo.h	13’ 48”	1’ 5”
TClass.h	13’ 13”	9’ 36”
TFile.h	3’ 15”	2’ 34”
TH1.h	4’ 25”	3’ 19”

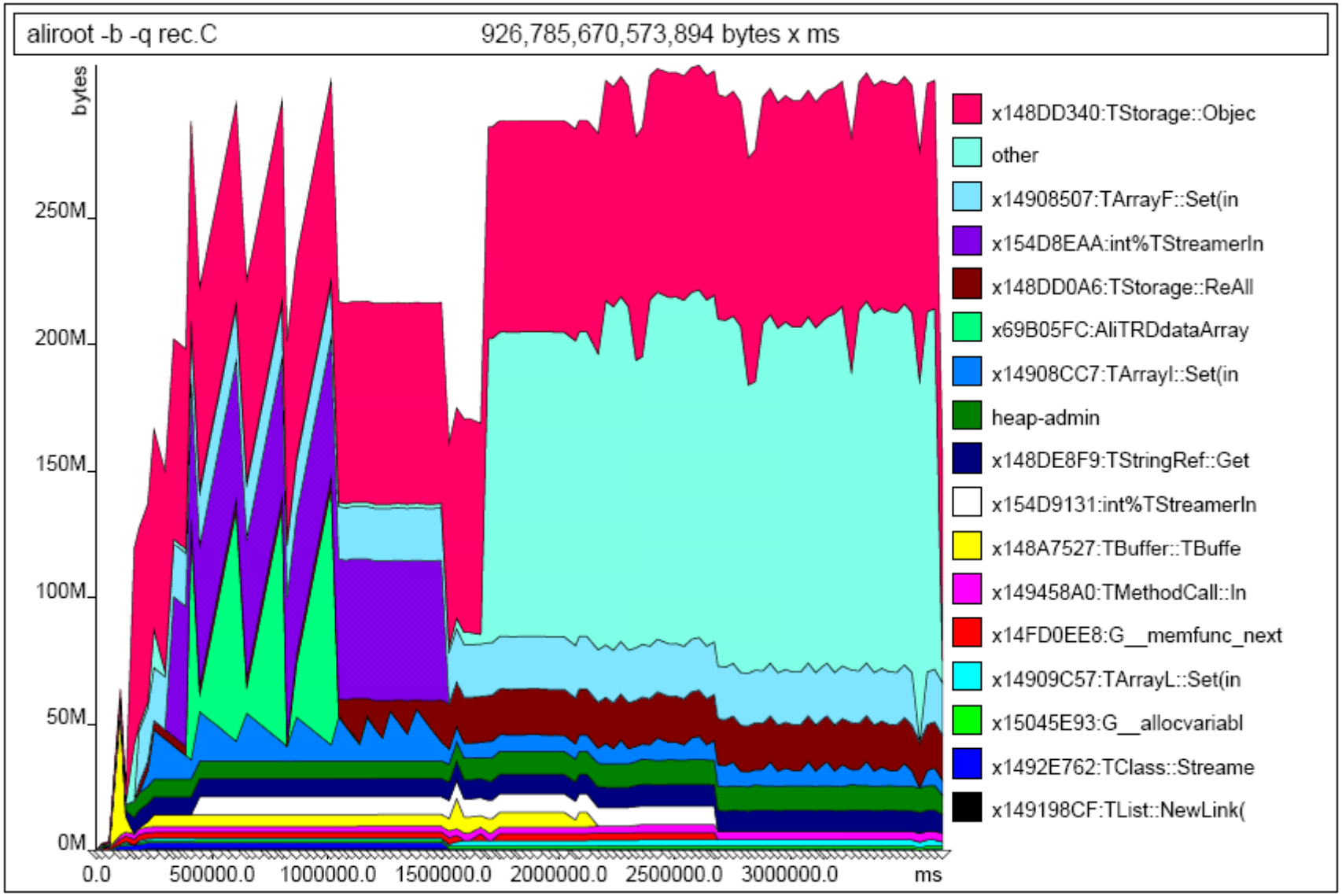
Heap size in 5.15/04



Alice AliRoot with root5.14

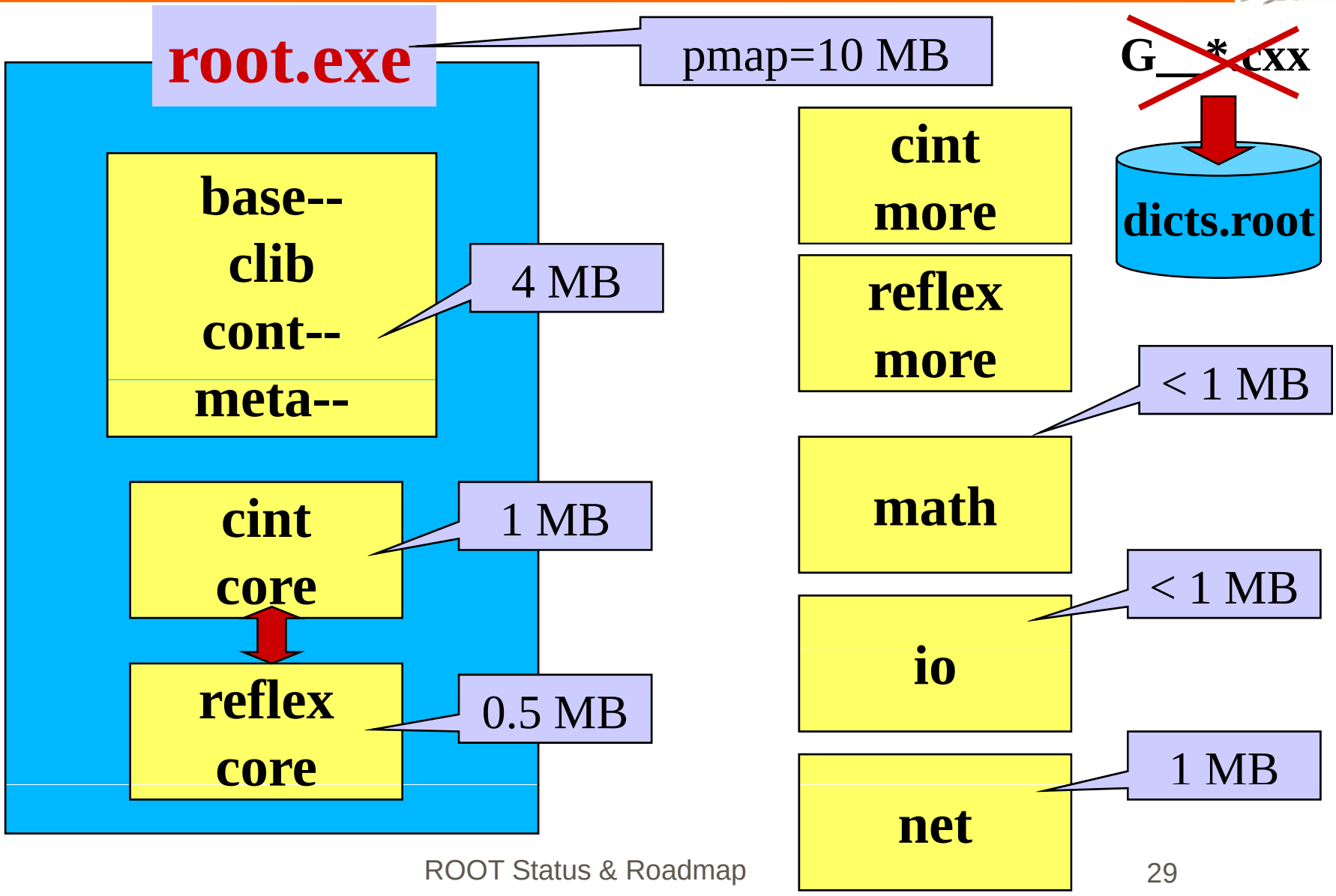


Alice AliRoot with root5.15/04



Libraries reorganization

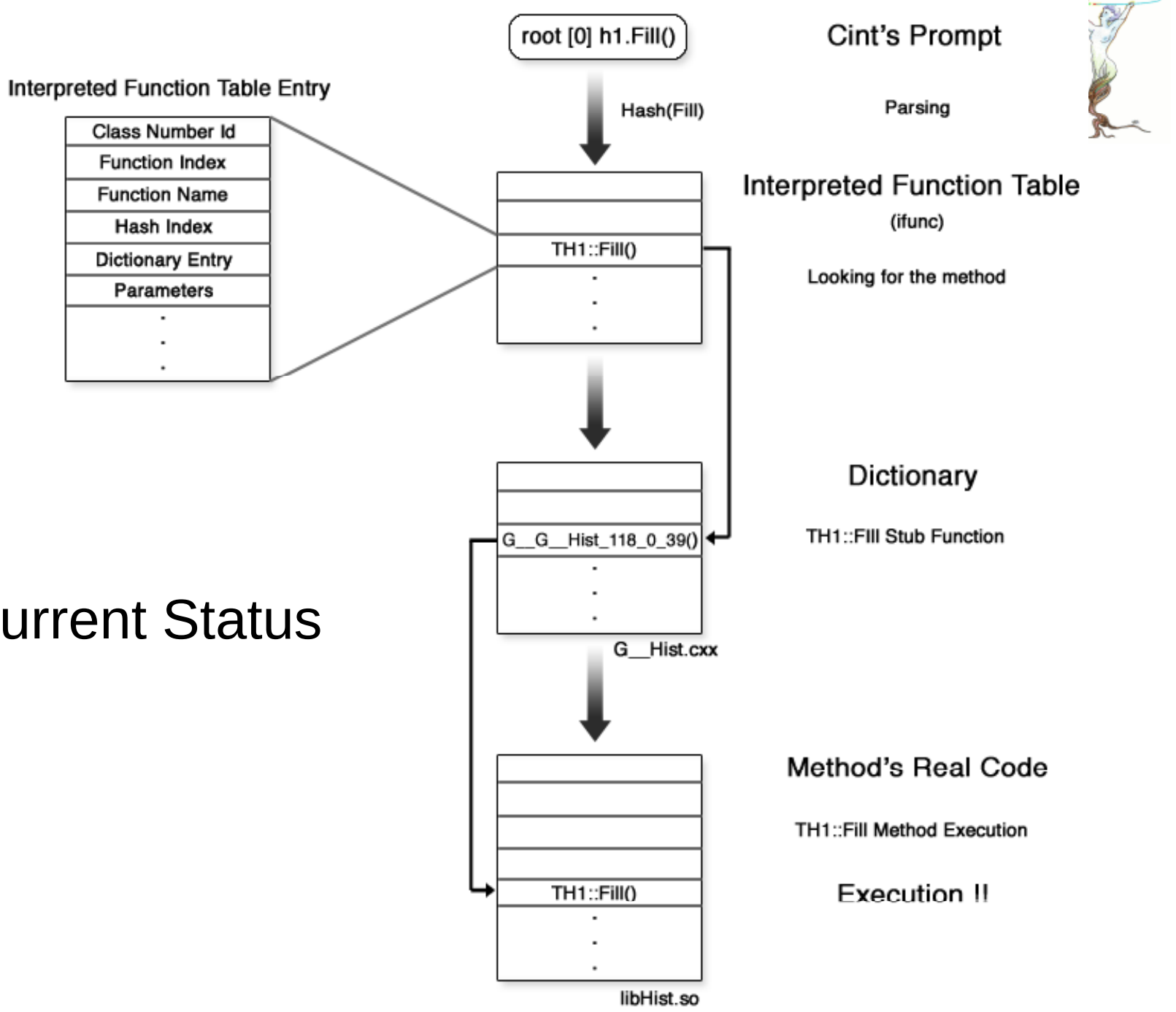
(proposal phase2 == BOOT phase1)

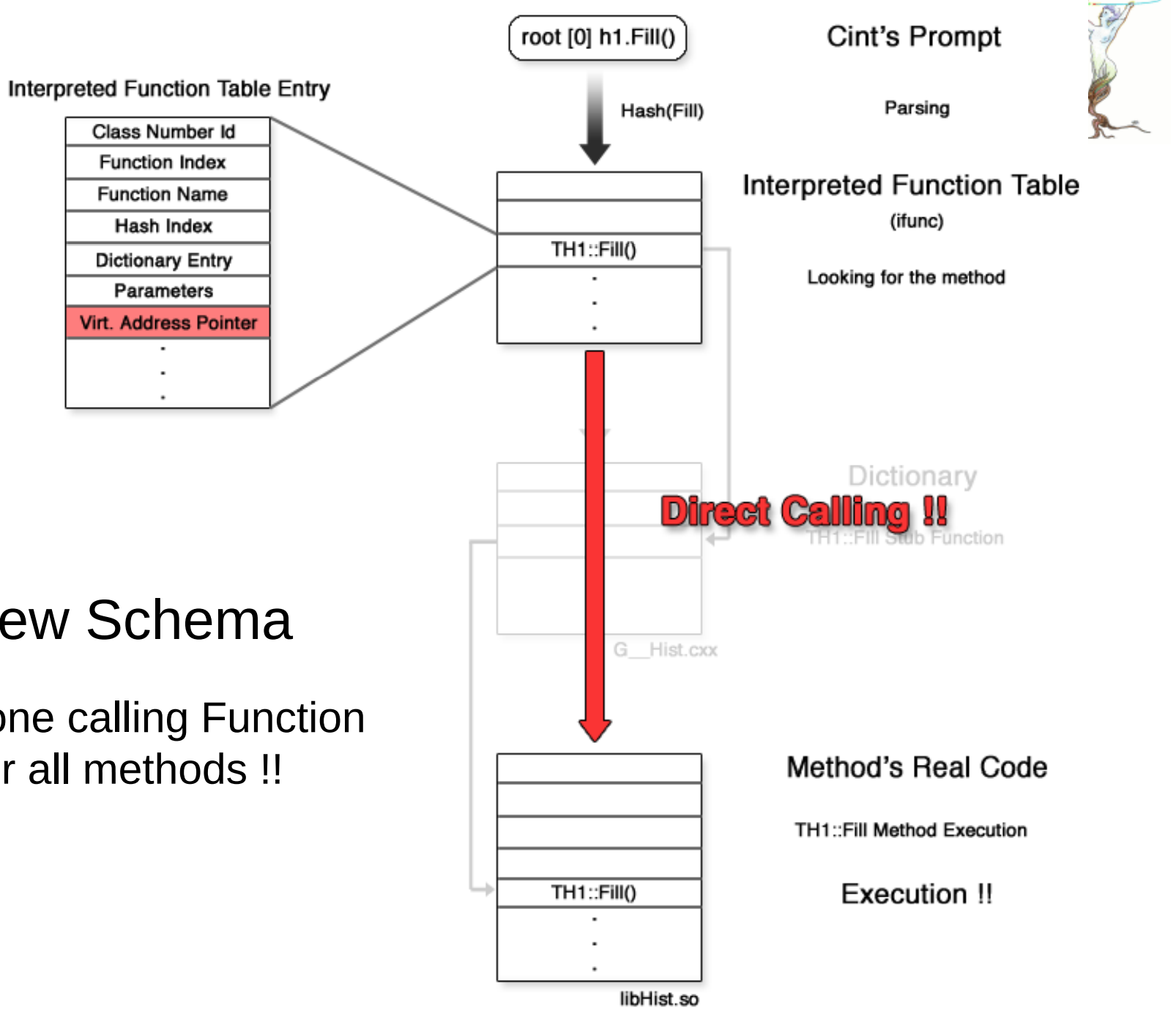


Reducing Dictionaries Size
(Current work by Diego & Leo)

- We are considering 3 phases
  - Eliminate the stub functions for all virtual functions, except the base definition. The gain in size is about 20 per cent.
  - Short-circuit the stub functions. See gain in coming slides (> 50%)
  - Eliminate a very large part (say 90%) of the generated code in dictionaries by storing the result of the rootcint/gccxml parsing directly into ROOT files (will be more compact and granular)

Current Status





New Schema

Only one calling Function
for all methods !!

Bypassing CINT stub functions

- We have now a first implementation of CINT that does not require stub functions.
- This code is machine/compiler dependent and currently tested only with gcc3.4 .
- We are planning to introduce very soon this new feature for gcc installations and keep the current old system for the other platforms (hoping to get it working for all versions of gcc and VC++.
- Inline functions: a difficult problem.

Removing CINT stubs

DICTIONARY'S SIZE (G__*.cxx)		
	CURRENT VERSION	FUTURE VERSION
mathcore	39619 LOC	13338 LOC
base	83816 LOC	3393 LOC
physics	8412 LOC	3104 LOC
treeplayer	25002 LOC	11385 LOC
geom	61004 LOC	22230 LOC
tree	27563 LOC	10626 LOC
g3d	17720 LOC	7015 LOC
geompainter	3799 LOC	1377 LOC
graf	32118 LOC	11245 LOC
matrix	38599 LOC	16818 LOC
meta	16820 LOC	6775 LOC
hist	44007 LOC	21514 LOC
gl	26333 LOC	11982 LOC
gpad	12367 LOC	4331 LOC
histpainter	3492 LOC	1479 LOC
Total	440671 LOC	146612 LOC

Total Code Saving Factor
3

Gain in shared libs

ROOT'S DYNAMIC LIBRARIES		
	CURRENT VERSION	FUTURE VERSION
libMathCore.so	2228 Kb	1020 Kb
libCore.so	7264 Kb	4888 Kb
libPhysics.so	596 Kb	376 Kb
libTreePlayer.so	1756 Kb	1244 Kb
libGeom.so	3856 Kb	2532 Kb
libTree.so	1944 Kb	1396 Kb
libGraf3d.so	1144 Kb	796 Kb
libGeomPainter.so	320 Kb	232 Kb
libGraf.so	2712 Kb	2000 Kb
libMatrix.so	2416 Kb	1816 Kb
libHist.so	2700 Kb	1932 Kb
libRGL.so	2480 Kb	1996 Kb
libGpad.so	932 Kb	676 Kb
libHistPainter.so	492 Kb	416 Kb
libMinuit.so	480 Kb	420 Kb
Total	28608 Kb	19740 Kb

Average Space Reduction Factor
1,44

Space Saving	
15 Root Libraries	8,8 MB

Total Space Saving Estimation	
The whole set of Libraries	17,2 MB

Next steps

- In the medium/long term, we would like to remove the generated code in dictionaries and replace it by the corresponding information in ROOT files.
- See coming talk about BOOT.

Conclusion

- The ROOT kernel classes have been reorganized to obtain a better granularity and save memory.
- Consolidation of the I/O sub-system.
- Important developments in all work packages described in the coming talks.
- Substantial reduction of the dictionaries in view with important gains in memory size.
- Other important developments facilitating the installation and use will be described in the talk about BOOT.

Acknowledgements

- The progress in the development in ROOT is possible thanks to
  - the strong support from the CERN management
  - The feedback from a huge and growing user community
  - The hard work and strong motivation of my colleagues in the project.

Thanks to all of you
And enjoy the workshop