WikiTrust

Experience is what you get...

Bo Adler
Ian Pye
Luca de Alfaro

ISSDM Symposium
UC Santa Cruz
October 2009

BIG(?) DATA

3M articles

200M revisions

around 1.5TB data

Some of you are probably rubbing your hands, thinking “my 1000 node cluster can eat that for breakfast.” This is a pretty big data set to handle, compared to what is usually tackled by university students. Not so big compared to what LANL, Google, or Facebook deal with at times. Wikipedia is struggling to handle this data; there have been no dumps of enwiki available since Jan 2008!

WikiTrust Version 1

Mediawiki stores data in mysql

20 days for us to load up!

Processing took over 10 days

Dealing with mysql was a pain. We RAIDed up our machine, and struggled with learning the mediawiki tools.

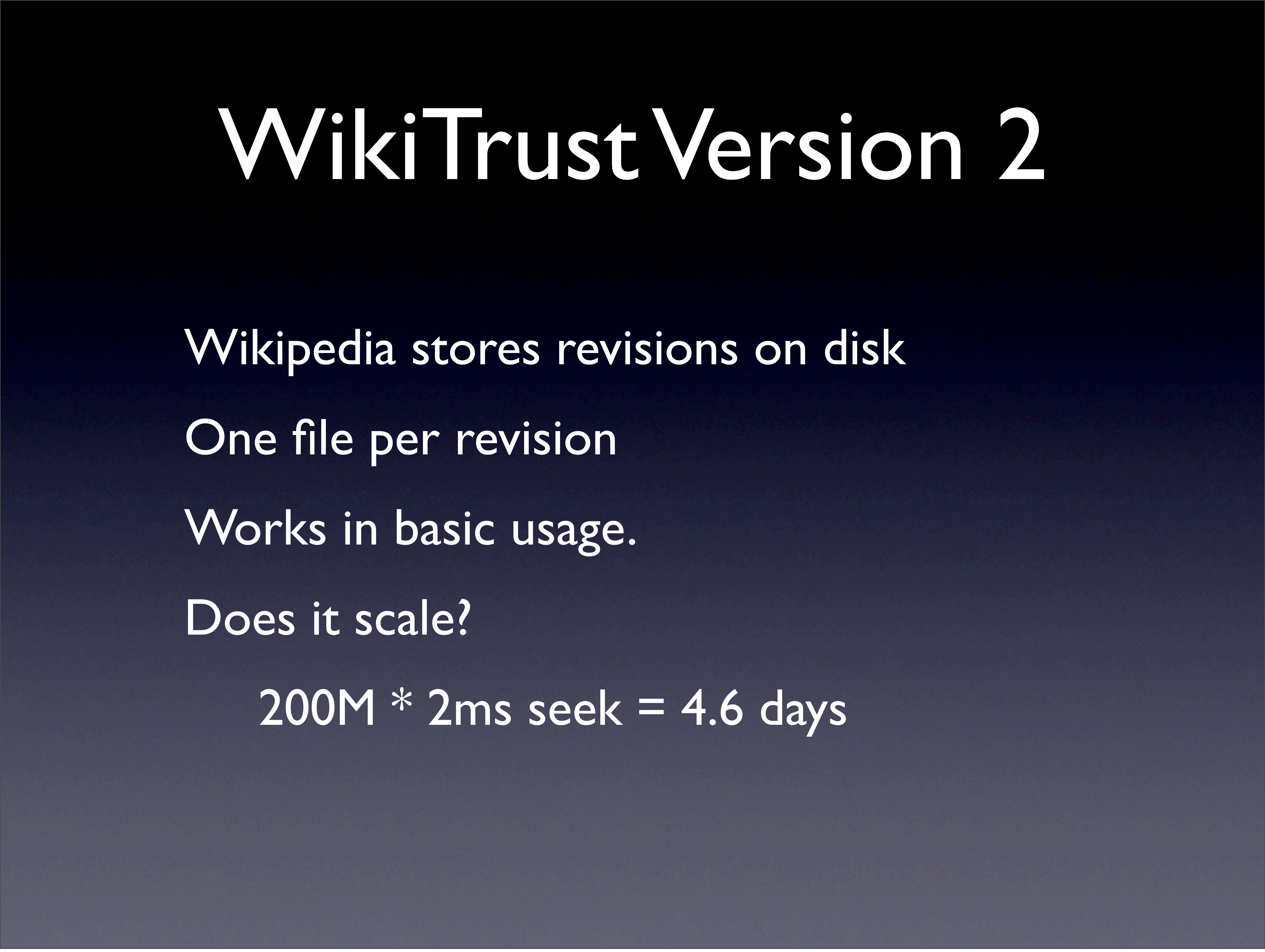
One big drawback of such a large data set is that doing any kind of processing takes days. (Remember those days of submitting batch jobs, where it was better to debug on paper than by IDE or printf?)

Turns out, Wikipedia doesn't store the revision text in mysql. D'oh!

WikiTrust Version 2

- Wikipedia stores revisions on disk
- One file per revision

We rewrote our code. Debated a few architectures, and decided that the easiest to code was to create one file per revision.



WikiTrust Version 2

Wikipedia stores revisions on disk

One file per revision

Works in basic usage.

Does it scale?

  200M * 2ms seek = 4.6 days

We did some testing, and everything seemed fine.

And then we had to rerun a computation, which revealed a problem with this organization.

WikiTrust Version 3

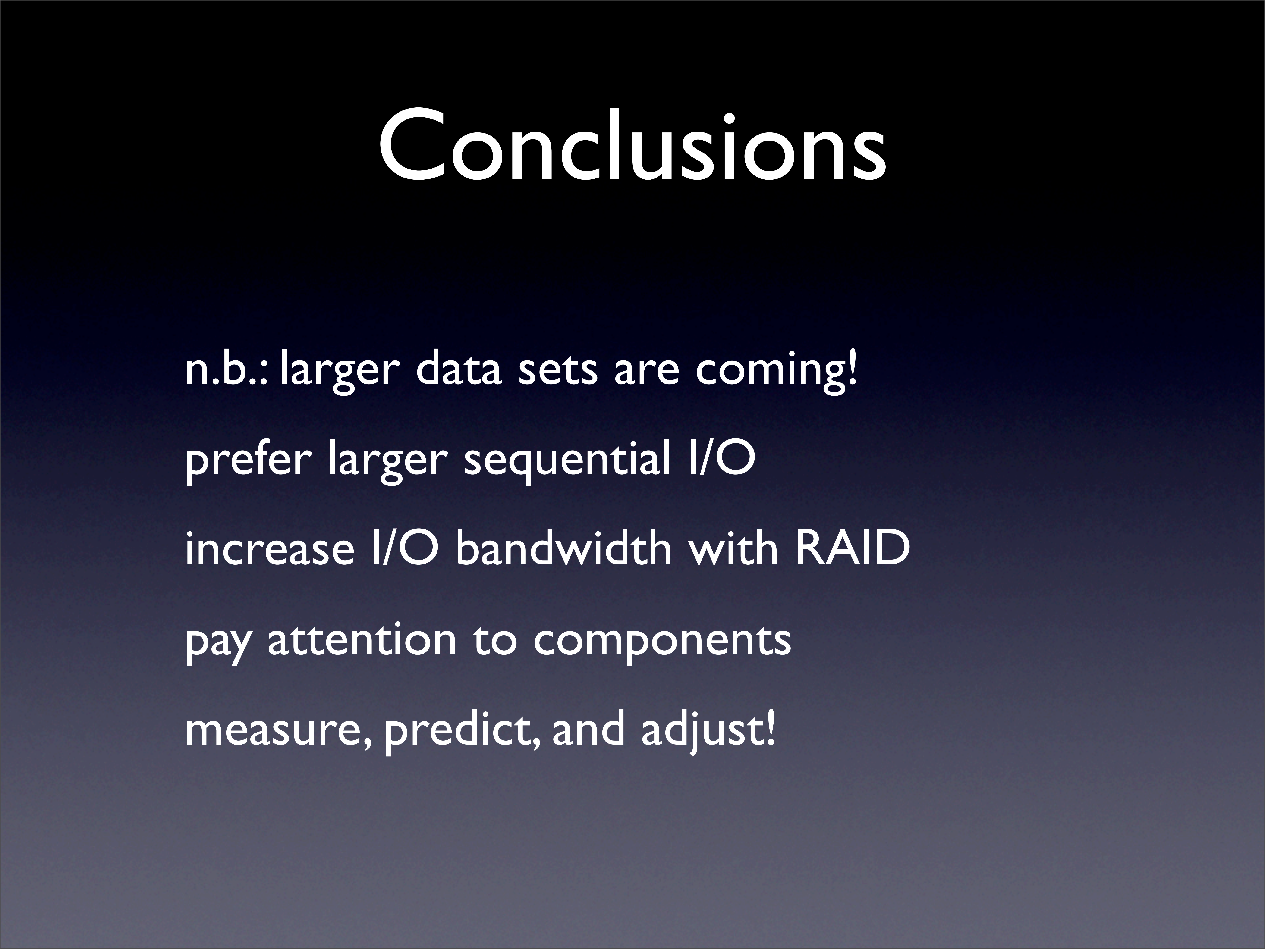
Now puts multiple revs per file.

Still have metadata in mysql.

Started optimizing component systems.

Another rewrite of the code, and now we can put the text of multiple revisions into a single file. When we are processing, this becomes much more efficient since we can read the file sequentially (and take advantage of read ahead).

XFS tip: mount as “noatime, nobarrier”
MySQL tip: use O_DIRECT on XFS
MySQL tip: separate log disk from data disk
Be careful about “nobarrier” since it might corrupt your data.



Conclusions

n.b.: larger data sets are coming!

prefer larger sequential I/O

increase I/O bandwidth with RAID

pay attention to components

measure, predict, and adjust!

Performance matters!

Useful Linux tools for measuring:
* top, vmstat iostat

Resources:
* RAID: http://www.percona.com/ppc2009/PPC2009_iotuning.pdf
* iostat: http://mituzas.lt/2009/03/11/iostat/
* XFS “nobarrier”: http://xfs.org/index.php/XFS_FAQ
* MySQL: http://www.mysqlperformanceblog.com/2007/11/01/innodb-performance-optimization-basics/

One more thing...

Do you speak... Italian? Portuguese? English?

We have a Firefox extension which lets you use our system. Today. On the live Wikipedia.
https://addons.mozilla.org/en-US/firefox/addon/11087

The English version is more of a functional "demo"; we are still trying to work out how to handle such a large data set and huge user base. But the Italian Wikipedia works great!

Demo

Thanks!

To keep up, follow me on Twitter: @thumper17