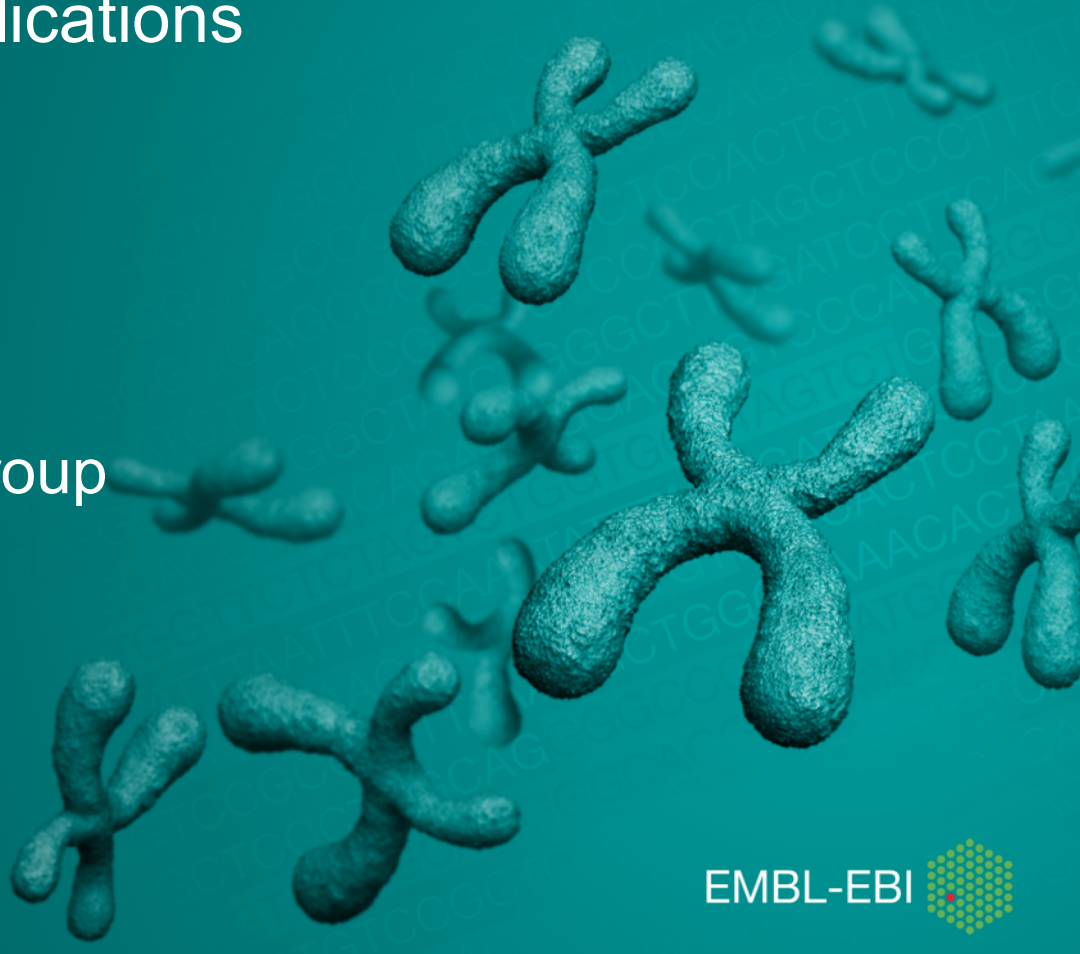
GPFS at EBI

Facing performance degradation when using mmap based applications

Jordi Valls

Systems Infrastructure Group

jvalls@ebi.ac.uk



EMBL-EBI

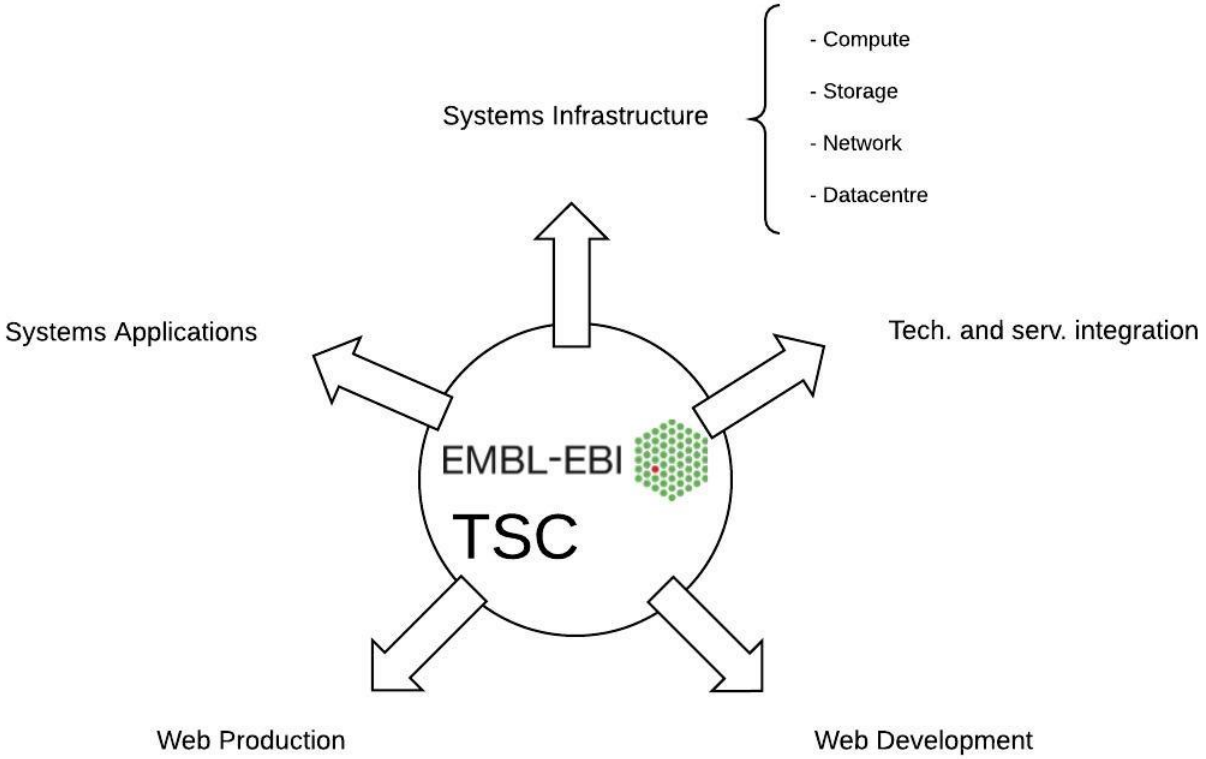
1. Who are EBI?

Europe's home for biological data, research and training

- A trusted data provider for life sciences.
- Part of the European Molecular Biology Laboratory (EMBL), an intergovernmental research organisation.
- International: 23 member states and 2 associate member states, 6 sites in Europe.
- EMBL-EBI manages a vast amount of public biological data, delivering it to the global life science community on demand, 24/7 without restriction or charge.

2. Systems at EBI

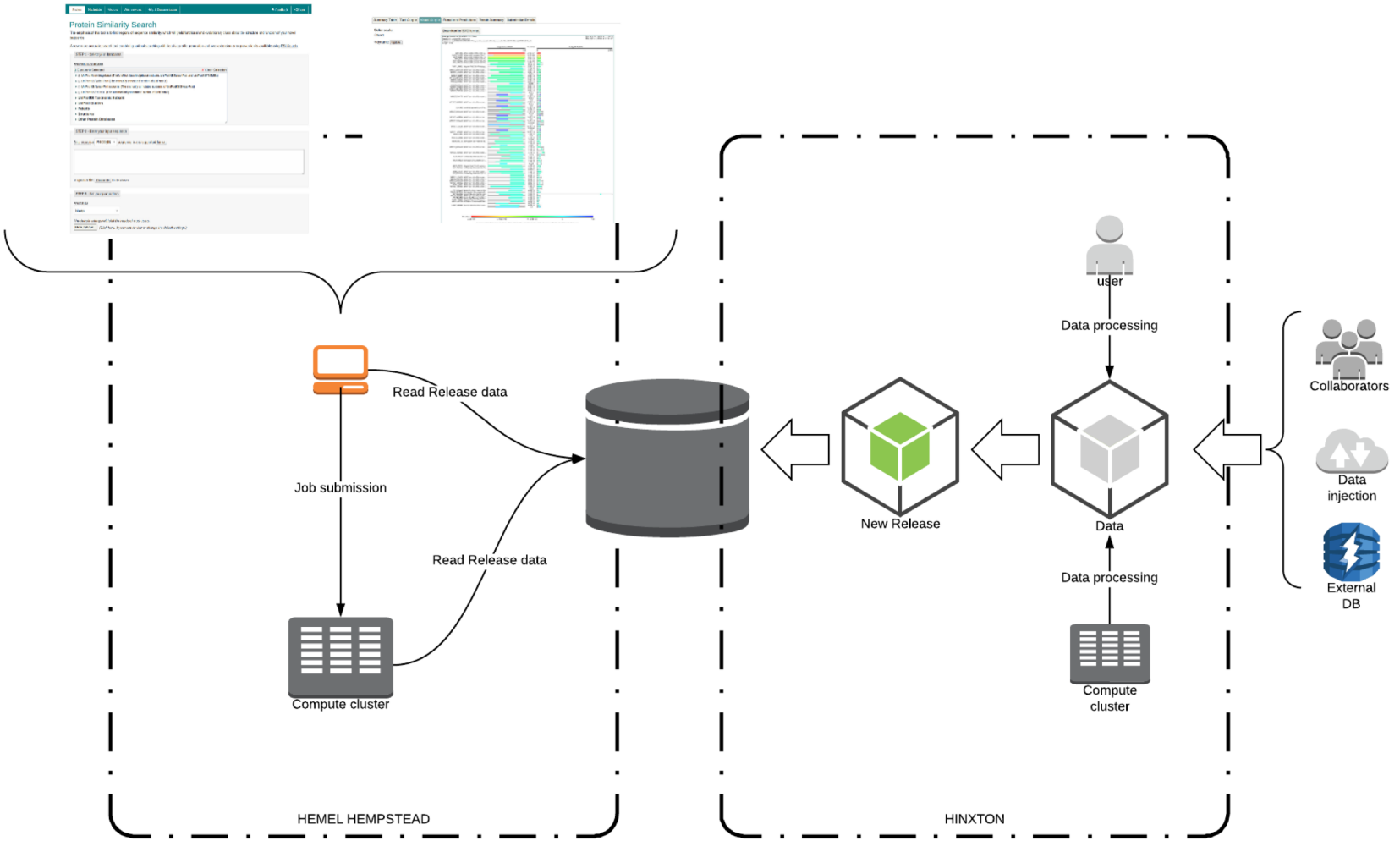
IT services at EBI are managed by the Technical Services Cluster (TSC)



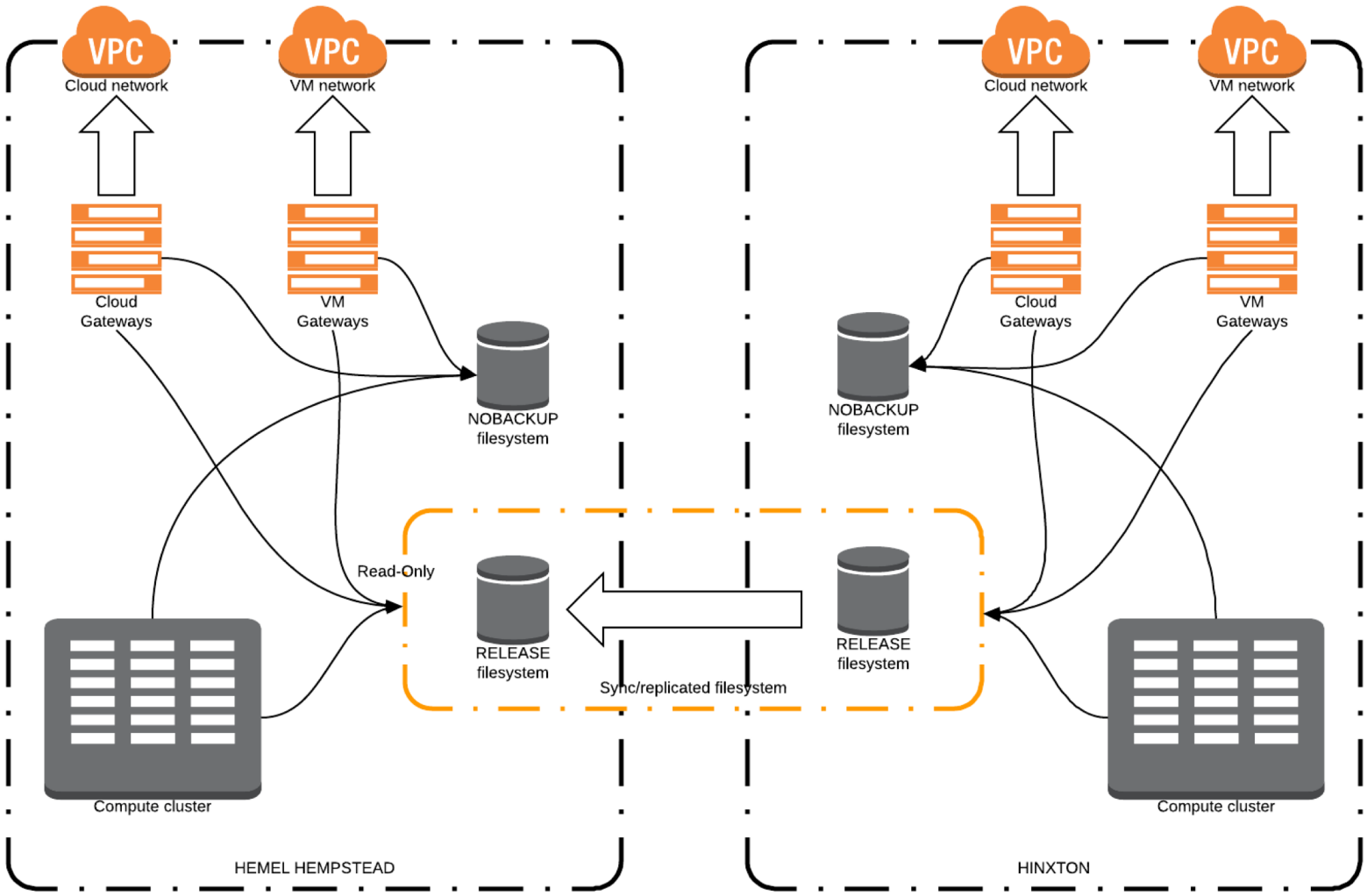
3. Infrastructure

- Three datacentres with around 200 racks of equipment.
- More than 40.000 cores. Mostly ethernet based.
  - Multiple HPC-like clusters (mainly blades).
  - Hadoop cluster.
- Over 140 PB of raw storage capacity.
  - Multiple technologies: NFS, Object Store, GPFS, Lustre
  - Annual storage growth of 40-50%

4. Scenario for GPFS: EBI services



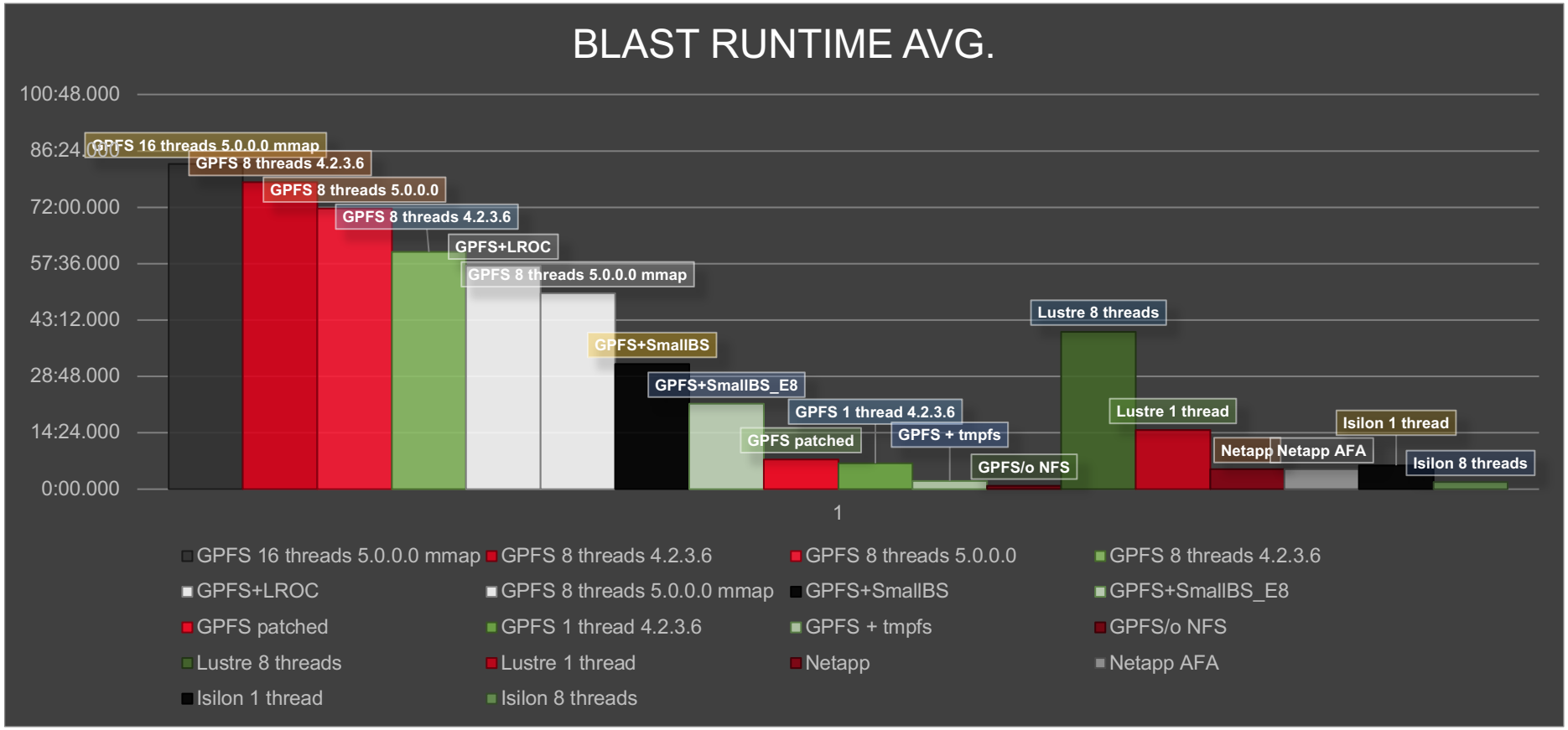
5. Scenario for GPFS: infrastructure



6. The problem: BLAST is slow

- BLAST (Basic Local Alignment Search Tools) is a software used to find regions of similarity between biological sequences.
- Developed by the NCBI and very popular in bioinformatics.
- Users reported running BLAST was extremely slow when source data was stored in GPFS.
- Some executions showed a performance 80 times slower compared to NFS.

6. The problem: BLAST is slow



7. First approaches

We tried multiples approaches without success.

- Blame the users.

7. First approaches

We tried multiples approaches without success.

- x Blame the users.
- Recompile BLAST using all optimization flags.

7. First approaches

We tried multiples approaches without success.

- x Blame the users.
- x Recompile BLAST using all optimization flags.
- Install the latest version of BLAST.

7. First approaches

We tried multiples approaches without success.

- x Blame the users.
- x Recompile BLAST using all optimization flags.
- x Install the latest version of BLAST.
- Update the GPFS client to the latest available version (5.0.0.1).

7. First approaches

We tried multiples approaches without success.

- x Blame the users.
- x Recompile BLAST using all optimization flags.
- x Install the latest version of BLAST.
- x Update the GPFS client to the latest available version (5.0.0.1)
- Use NVMe, either using LROC or as a filesystem. (thanks Gurdip and E8).

7. First approaches

We tried multiples approaches without success.

- x Blame the users.
- x Recompile BLAST using all optimization flags.
- x Install the latest version of BLAST.
- x Update the GPFS client to the latest available version (5.0.0.1).
- x Use NVMe, either using LROC or as a filesystem. (thanks Gurdip and E8).

8. The real evil: mmap

BLAST switches to using mmap for IO when it's executed with more than 1 thread.

- Performance degradation only occurs when running **multiple threads** in the same host.
- Analizing the execution of BLAST using strace we realized there were thousands of **mmap calls**.
- Many users reporting problems with parallel filesystems when using mmap, including GPFS.
- mmap causes IO to fallback to the default kernel page size (4K) instead of using the values defined for GPFS.

8. The real evil: mmap

BLAST switches to using mmap for IO when it's executed with more than 1 thread.

```
=== mmdiag: iohist ===

I/O history:

 I/O start time RW    Buf type disk:sectorNum     nSec  time ms  Type    Device/NSD ID        NSD node
--------------- -- ----------- ----------------- -----  -------  ---- ------------------ ---------------
16:34:53.284905  R        data    2:338059951504     8   17.235   cli AC110407:586CEE7B       10.7.74.15
16:34:53.302271  R        data    2:338059912744     8    4.659   cli AC110407:586CEE7B       10.7.74.15
16:34:53.307139  R        data    1:338060022688     8    1.374   cli AC110406:586CEE77       10.7.74.14



=== mmdiag: iohist ===

I/O history:

 I/O start time RW    Buf type disk:sectorNum     nSec  time ms  Type    Device/NSD ID        NSD node
--------------- -- ----------- ----------------- -----  -------  ---- ------------------ ---------------
16:24:17.783564  R        data    2:214062309376  4096   22.108   cli AC110407:586CEE7B       10.7.74.15
16:24:17.806245  R        data    1:214062120960  4096   23.263   cli AC110406:586CEE77       10.7.74.14
16:24:17.830125  R        data    1:214062116864  4096   18.072   cli AC110406:586CEE77       10.7.74.14
```

8. The real evil: mmap

BLAST switches to using mmap for IO when it's executed with more than 1 thread.

File `objtools/blast/seqdb_reader/seqdbimpl.hpp`

```
/// Set number of threads
///
/// Set number of threads which will share this object.  This
/// permits use of an internal mmap for the threads.
/// If the second parameter is 'false' (the default),
/// the internal mmap is not used if num_threads == 1.
/// For certain applications where there are multiple CSeqDB
/// objects, each one accessed by only a single thread,
/// setting num_threads to 1 (thread per CSeqDB) results in
/// a performance hit by not using the mmap.
/// In this case, force_mt ("force multithreading") should
/// be set to 'true' to allow use of the mmap when num_threads
/// == 1.  For num_threads > 1, force_mt has no effect.
///
/// @param num_threads The number of threads which will share
///                    access to this CSeqDB object. [in]
/// @param force_mt Defaults to false, setting to true when
///                 num_threads == 1 forces multithread
///                 internal mmap. [in]
void SetNumberOfThreads(int num_threads, bool force_mt = false);
```

9. Next step: help IBM!

No workaround could be provided:

- Running BLAST with 1 thread takes too much time.
- Switching to an MPI solution was too complex.

So we looked for help:

- Asked on the GPFS Users Group list
- Contacted with Sven Oehme
- Sven agreed on working on a fix for the problem


Thank
You