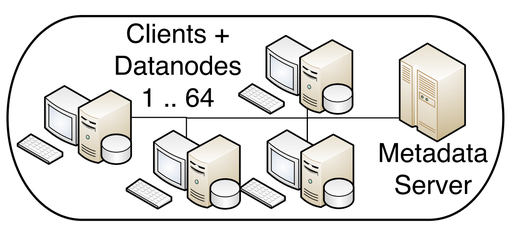
Mitigating the Network Impact in Large Scale DFSs

Gustavo Bervian Brand, Adrien Lèbre
SCALUS EU Project, Mines de Nantes / INRIA / LINA - FRANCE

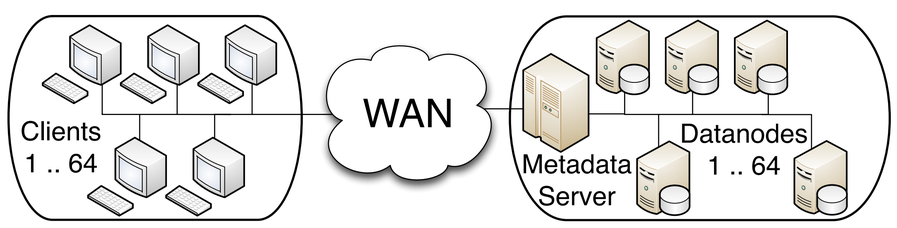
Most DFSs rely on a static model that does not take into account the scope and the requirements of each application w.r.t the physical topology where they are deployed. However, considering the increasing trend of using multiple sites to share data across applications, we propose the investigation of a new model of DFS that consider LAN vs WAN traffic in order to mitigate the performance impact of the network exchanges.

Network Topology Impact

- **Is it still relevant to use additional nodes to analyze data faster**?
- **How applications accessing data through the DFSs at LAN/WAN levels are impacted**?
- **Should locality be considered as a major concern to design a DFS**?
- We investigated these points by conducting several experiments with HDFS and Lustre where Clients (C), Datanodes (D) and a Metadata server (M) are deployed on a node, across a LAN ("-") or a WAN ("/").



CD-M



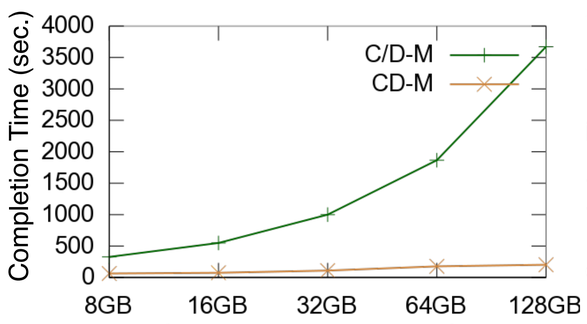
C/D-M

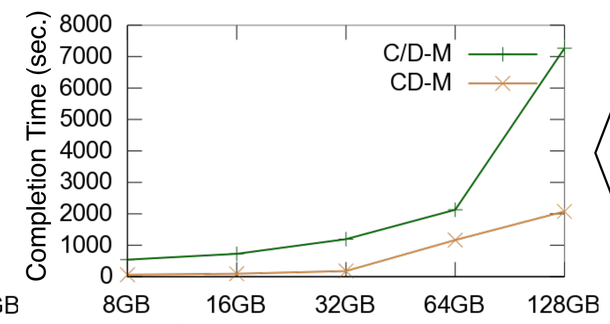
CD-M: client and datanode are at the same node, while the metadata server is reachable at the LAN
C/D-M: clients are separated from the datanodes and the metadata server through a WAN

- The applications used were **Hadoop** based **Grep**, **Text-writer** and **Sort benchmarks**.
- The file size grew from 8GB to 128GB with up to 64 nodes at the Grid'5000 testbed.

In most cases, accessing data through WAN leads to worse performance.
It was better to use less nodes than trying to benefit from external WAN ones.

- The completion time for the local scenarios with 16 nodes (CD-M and C-D-M) is similar to the WAN ones (CD/M, C-D/M, C/D-M, CD-M/CD and C-D-M/C) using 2x or 4x more nodes.



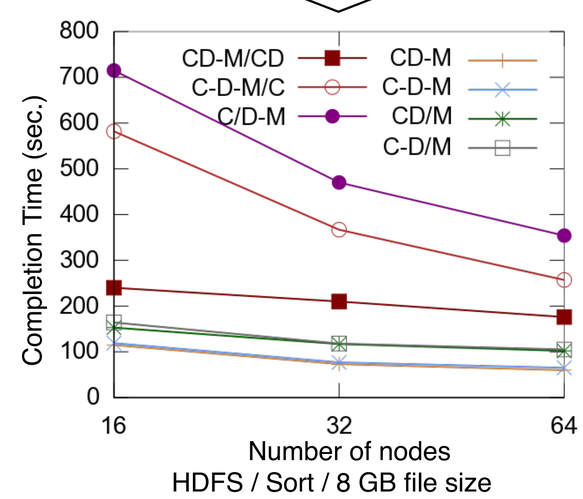


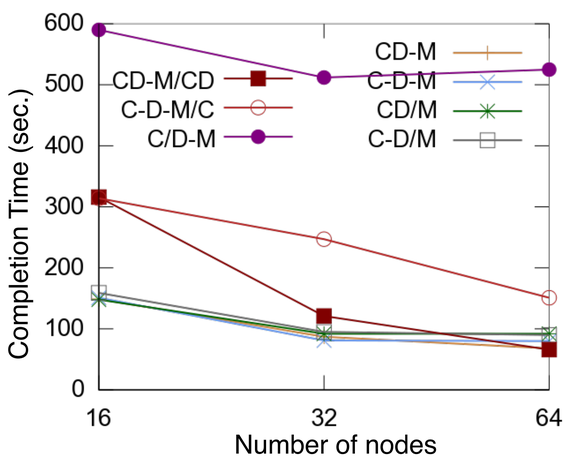
Sort Test	HDFS		Lustre	
File size	CD-M	C/D-M	CD-M	C/D-M
	Tests with 64 nodes			
8 GB	63	327	61	543
16 GB	75	549	93	726
32 GB	109	999	179	1194
64 GB	177	1866	1166	2129
128 GB	202	3671	2075	7267

Sort completion time
File sizes from 8 GB to 128 GB files

Systems	HDFS			Lustre		
Scenarios	Grep	Writer	Sort	Grep	Writer	Sort
	Tests with 16 nodes					
CD-M	81	61	**115**	110	54	149
C-D-M	83	66	**119**	114	48	151
CD/M	135	76	153	110	52	148
C-D/M	134	76	164	125	55	159
CD-M/CD	162	60	240	113	76	316
C-D-M/C	116	104	582	201	114	314
C/D-M	169	408	715	345	194	590
	Tests with 32 nodes					
CD-M	76	45	73	89	41	87
C-D-M	76	57	77	89	45	81
CD/M	121	62	**117**	88	41	92
C-D/M	122	63	**118**	99	48	95
CD-M/CD	136	57	**210**	89	55	121
C-D-M/C	113	100	**367**	149	105	247
C/D-M	136	245	**470**	275	175	512
	Tests with 64 nodes					
CD-M	68	44	60	73	36	68
C-D-M	82	62	65	90	56	80
CD/M	110	63	**102**	80	36	92
C-D/M	127	72	**105**	101	60	90
CD-M/CD	107	51	**176**	79	38	66
C-D-M/C	116	94	**257**	182	117	151
C/D-M	141	224	**354**	272	176	525

Completion time of the tests with 8 GB file size





Drawbacks of Current DFSs

- **Elasticity is a new aspect of distributed systems, consisting on using external resources at any time to compute and process data faster.**

- Do multi-applications environments increase the "locality" concern?
- Why "local scope" applications should suffer the penalty of external server communications in charge of managing data or metadata?
- Why data should be pushed over the network if it will be used only locally, or even worse, simply deprecated by the end of the application's execution?
- Why data needs to be pushed from one location to another, instead of using an on-demand pulling model?
- Can we consider the physical topology to improve the performance as well the scalability of DFSs (like HDFS does for reliability concerns)?

Work in Progress

Next DFSs should consider *the physical topology* and *the application's scope* to prevent performance impacts from the network exchanges. By avoiding *unsolicited traffic* as much as possible, we promote:

- ***Group wide striping mechanism***

  Data is spread according to the applications' access patterns (avoiding alignment concerns) across nodes belonging to the same group.

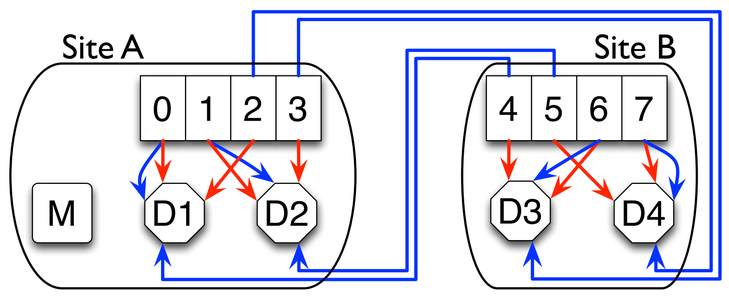
- ***Distributed persistent LRU cache mechanism***

  Once data has been pulled from one group to another one, it will never be pulled again unless if it has been modified on the other site ("group-wide LRU").

- ***Explicit reliability***

  Applications should explicitly mention the factor replica and how wide this data should be replicated (LAN or WAN).

Example of data blocks placement according to the applications behavior (CD-M/CD)



Round-Robin "infrastructure wide" striping strategy
Round-Robin "group wide" striping strategy

Contacts: Gustavo.Bervian-Brand@mines-nantes.fr
Adrien.Lebre@mines-nantes.fr
ASCOLA Research Group - SCALUS
http://www.scalus.eu
ECOLE DES MINES DE NANTES