Update on PROOF-Lite

G. GANIS
CERN / PH-SFT
for the ROOT team

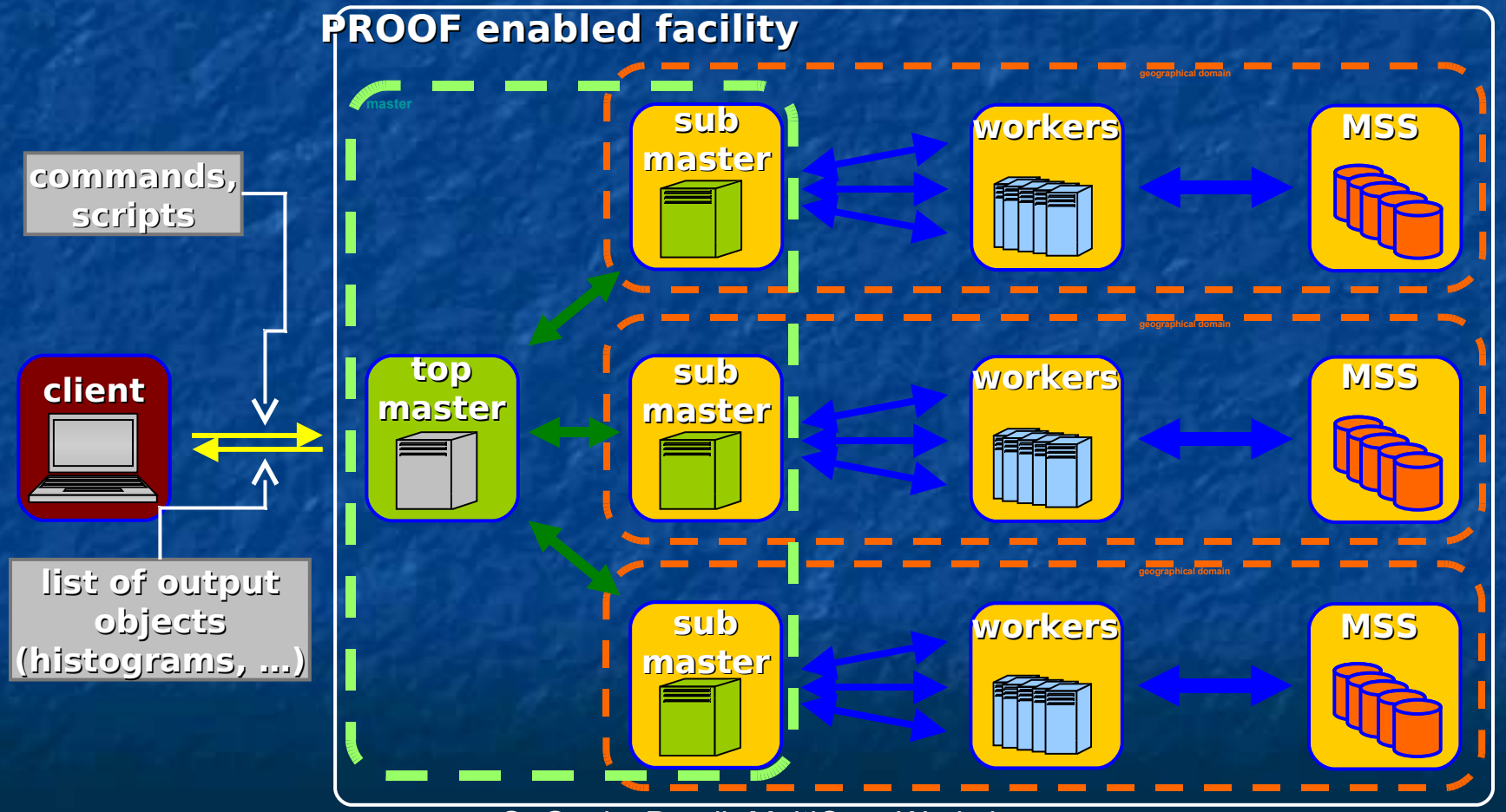
Workshop on Parallelization and MultiCore technologies for LHC, CERN, Oct 2008

Outline

- Reminder: PROOF, PROOF-Lite
- Optimizing IO
  - Parallel unzipping
  - SSD
- First look at with copy-on-write start-up
- Summary

PROOF in a slide

PROOF: Dynamic approach to end-user HEP analysis on distributed systems exploiting the intrinsic parallelism of HEP data

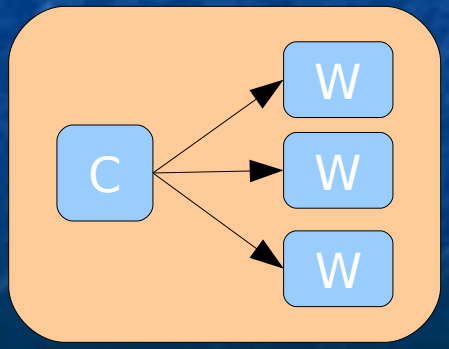


PROOF

- PROOF has been developed having in mind the case of T2/T3 analysis facilities, clusters O(100) nodes
- Its flexible multi-tier architecture allows to adapt to very different situations, and to move in size in both directions
  - Expand to federate clusters, eventually to the GRID
    - See A. Manarof at PROOF07
  - Shrink to few machines
- Multi-Core is at the extreme: one machine, lot of CPU power …

PROOF Lite

- PROOF Lite is a realization of PROOF in 2 tiers
  - The client starts and controls directly the workers
  - Communication goes via UNIX sockets
- No need of daemons:
  - workers are started via a call to ‘system’ and call back the client to establish the connection
- Starts N_{CPU} workers by default



PROOF Lite (2)

Additional reasons for PROOF-Lite

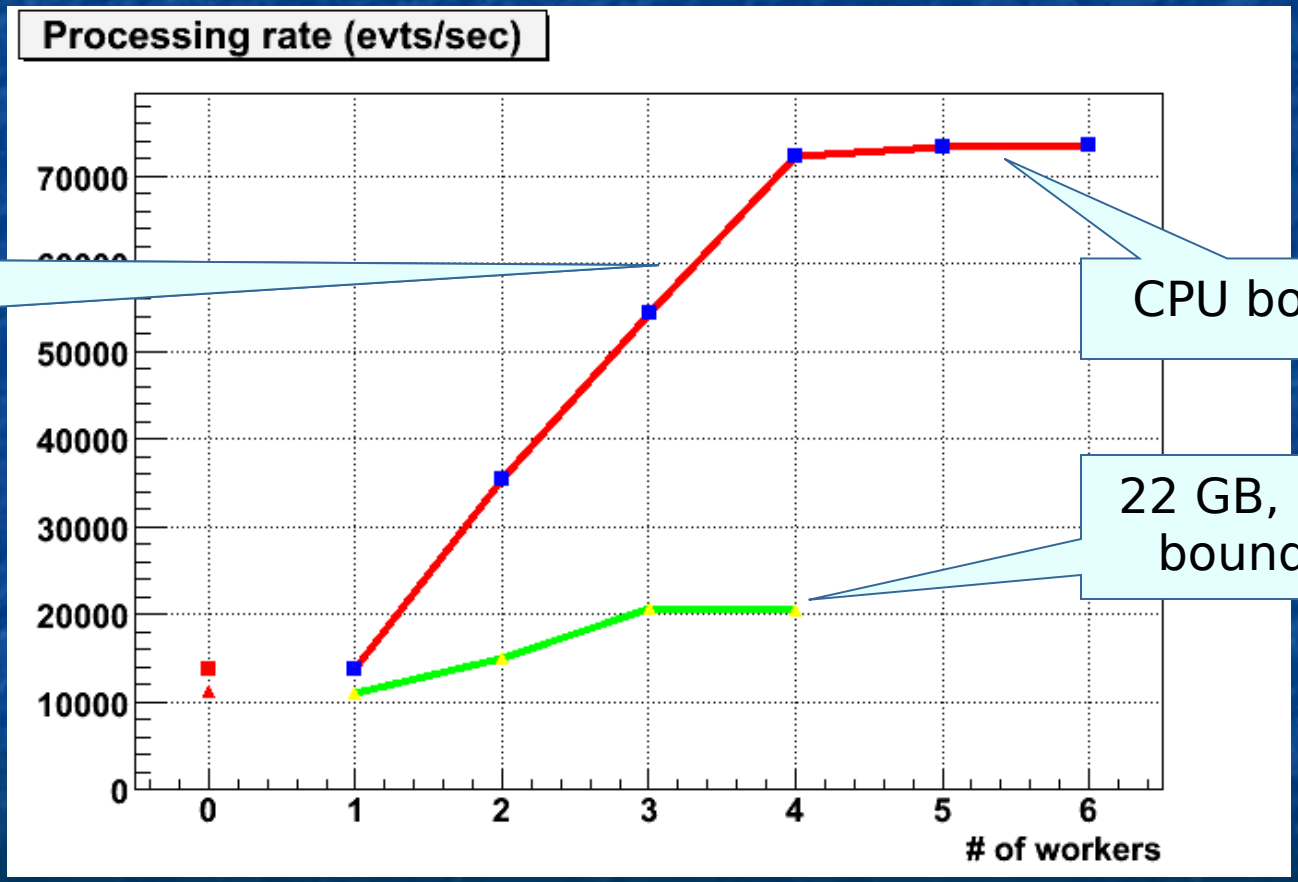
- Can ported on Windows
  - There is no plan to port current daemons to Windows
  - Needs a substitute for UNIX sockets
    - Use TCP initially
- Can be easily used to test PROOF code locally before submitting to a standard cluster
  - Some problems with users' code are difficult to debug directly on the cluster

PROOF-Lite performance

- CPU-bound
  - already quite good
- IO-bound
  - critically depends on I/O performance as for all systems

Scaling processing a tree, example

- Data sets 2 GB (fits in memory), 22 GB



2 GB, no memory refresh

CPU bound

22 GB, IO bound

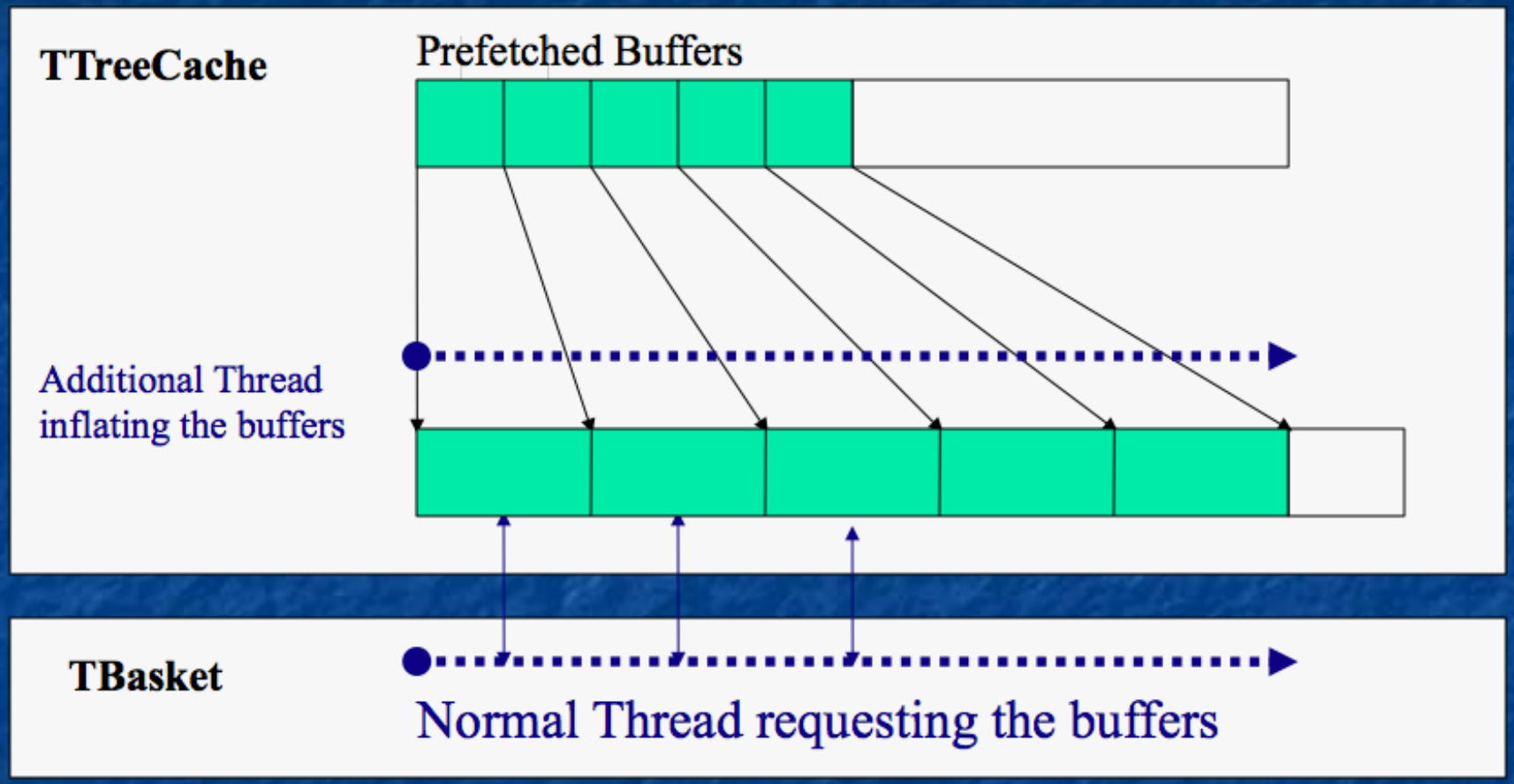
4 cores, 8 GB RAM, single HDD

Using cores to improve IO

- When reading data a large fraction of time is spent in decompressing
- This a case where additional core(s) may help and it is a dedicated task under control of ROOT which could already be done now in a separated thread
- Now available in ROOT

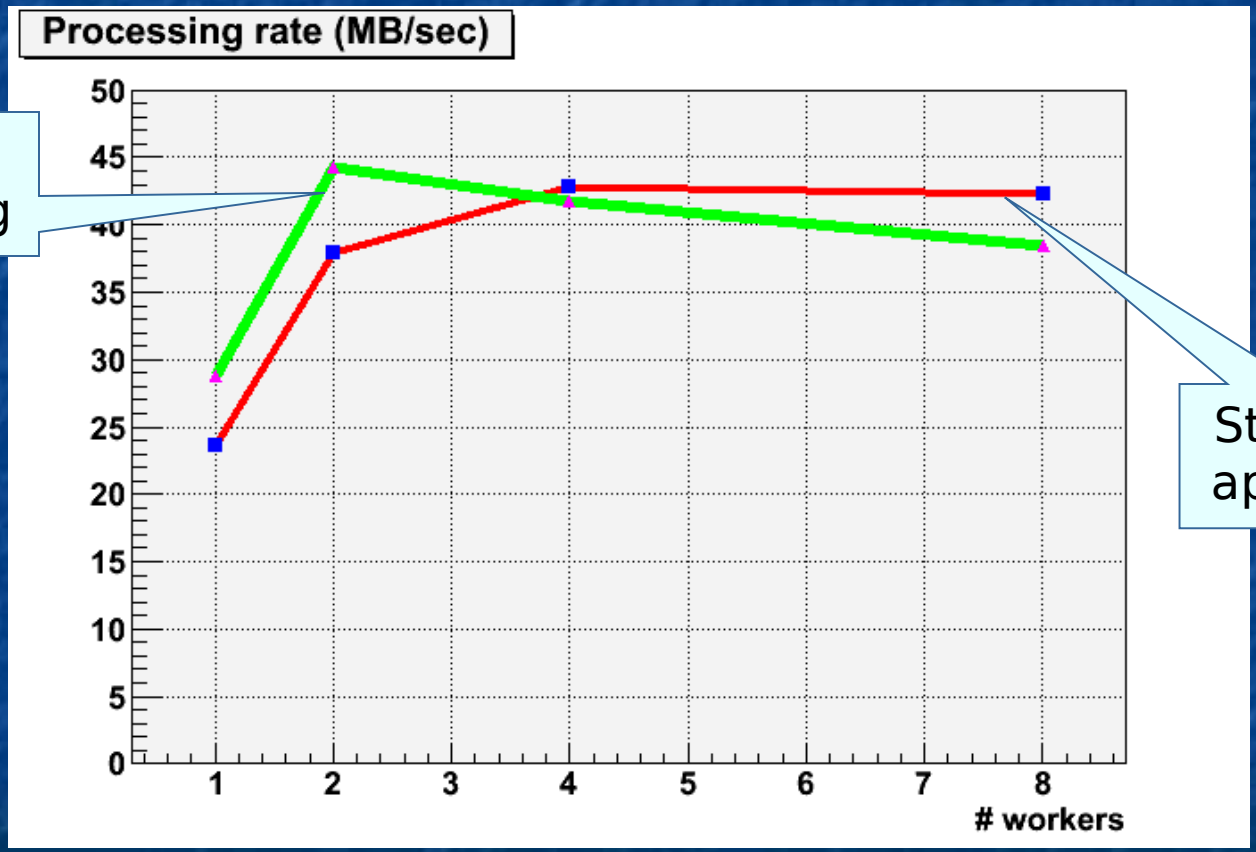
Basic Idea

Courtesy of L. Franco



Results on a 8 core machine

- 8 cores, 16 GB RAM, 2.5 TB under RAID 5



Parallel unzipping: comments

- One thread per file
- ~20% improvement for less than 4 workers
- 4 workers is equivalent to about 8 processes
  - Above that there are more processes than cores with a slowly increasing negative interference

Solid State Disk Testing with PROOF

Courtesy of S. Panitkin, BNL

- BNL PROOF Farm Configuration for this test
- 10 nodes - 16 GB RAM each
- 80 cores: 2.0 GHz Kentsfields
- 5 TB of HDD space (10x500 GB)
- 640 GB SSD space (10x64 GB)

Solid State Disk @ BNL

Courtesy of S. Panitkin, BNL

- Model: Mtron MSP-SATA7035064
- Capacity 64 GB
- Average access time ~0.1 ms (typical HD ~10ms)
- Sustained read ~120MB/s
- Sustained write ~80 MB/s
- IOPS (Sequential/ Random) 81,000/18,000
- Write endurance >140 years @ 50GB write per day
- MTBF 1,000,000 hours
- 7-bit Error Correction Code

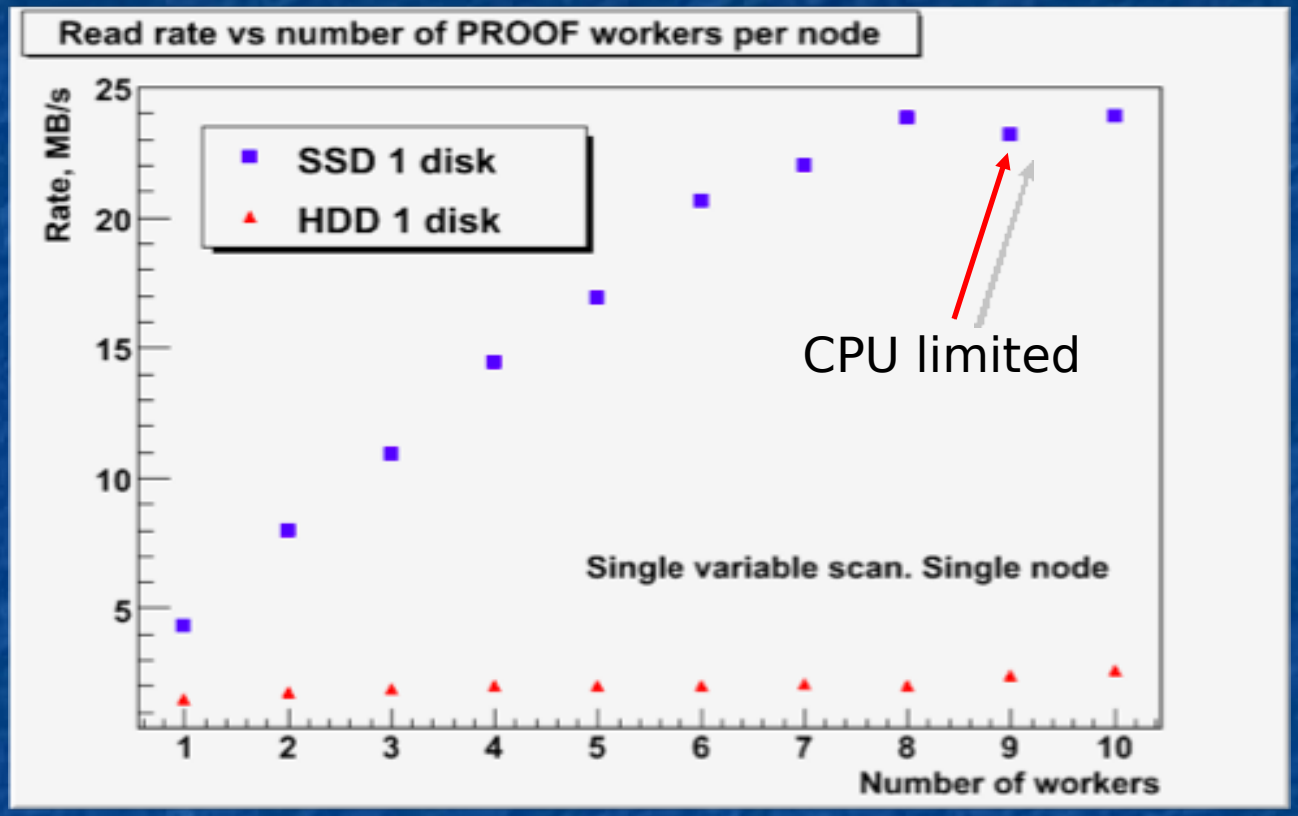
Test Configuration

Courtesy of S. Panitkin, BNL

- 1+1 or 1+8 nodes PROOF farm configurations
- 2x4 core Kentsfield CPUs per node, 16 GB RAM per node
- All default settings in software and OS
- Different configuration of SSD and HDD hardware depending on tests
- "PROOF Bench" suit of benchmark scripts to simulate analysis in ROOT.
  - Data simulate HEP events ~1k per event
  - Single ~3+ GB file per PROOF worker in this tests
- Reboot before every test to avoid memory caching effects
- This set of tests emulates interactive, command prompt root session
  - Plot one variable, scan ~10E7 events, ala D3PD analysis
- Looking at read performance of I/O subsystem

SSD versus HDD

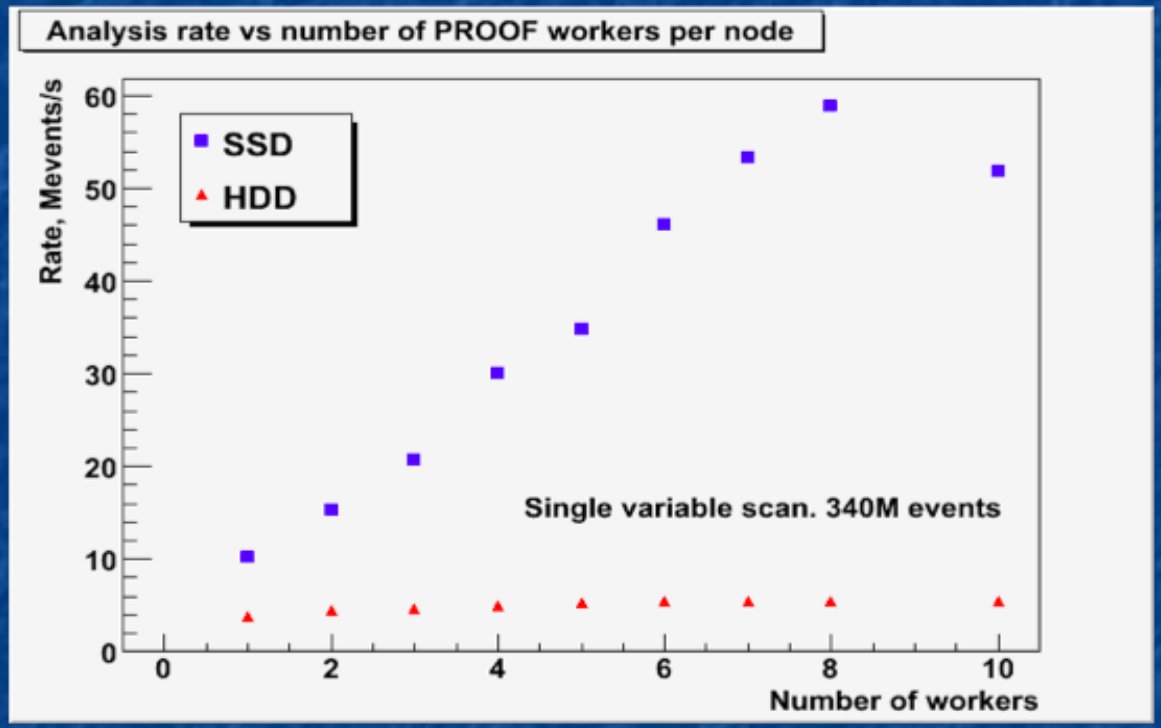
Courtesy of S. Panitkin, BNL



- SSD holds clear speed advantage
- ~ 10 times faster in concurrent read scenario

SSD vs HDD on 8 Node Cluster

Courtesy of S. Panitkin, BNL



- Aggregate (8 node farm) analysis rate as a function of number of workers per node
- Almost linear scaling with number of nodes

Startup optimizations: copy-on-write

- Useful in optimizing the memory print in the case of large input or condition data which is typically the same and constant for all the processes
- The fork() should occur at the very last moment, when all the settings are done
- In our prototype it happens just before calling Process(...)

Copy-on-write: PROOF-Lite prototype

- At startup only a worker out of N is started
- All the initializations are done, i.e. all the needed packages are loaded and the input list created
- When invoking Process(...), internally the controller (client/master) ask the single worker to clone N-1 times.

Copy-on-write: first numbers

- We see the effect on the memory foot print already in a very simple and light example
  - Standard

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4728	ganis	20	0	176m	41m	23m	S	0	2.1	0:03.58	root.exe
4738	ganis	20	0	108m	28m	15m	S	0	1.4	0:01.10	proofserv.exe
4740	ganis	20	0	108m	28m	15m	S	0	1.4	0:01.24	proofserv.exe
4743	ganis	20	0	108m	27m	15m	S	0	1.4	0:01.06	proofserv.exe
4750	ganis	20	0	108m	28m	15m	S	0	1.4	0:01.26	proofserv.exe

  - Copy-on-write enabled

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
9410	ganis	20	0	177m	41m	23m	S	0	2.1	0:03.48	root.exe
9420	ganis	20	0	108m	28m	15m	S	0	1.4	0:01.66	proofserv.exe
9434	ganis	20	0	108m	19m	7388	S	0	1.0	0:00.38	proofserv.exe
9435	ganis	20	0	108m	19m	7176	S	0	1.0	0:00.30	proofserv.exe
9436	ganis	20	0	108m	19m	7200	S	0	1.0	0:00.64	proofserv.exe

  - 50 MB saved out of 170 MB

Copy-on-write: PROOF

- Interesting also for standard PROOF
- Needs to start only 1 worker per machine, so that all the initializations are done
- Just before start procesing the additional workers are creating, possibly taking into account the load in the each node
- Ongoing work

Summary

- PROOF performances in processing trees benefit from IO optimizations both coming from smart software or from better hardware
  - Parallel unzipping may give 20% performance gain under certain circumstances
  - SSD technology can give O(10) factors wrt standard HDD
- Copy-on-write techniques for optimized worker startup are being prototyped on PROOF-Lite
  - First results encouraging
  - Integration into PROOF under study