Ellexus Ltd: The I/O Profiling Company

Dr Rosemary Francis, CEO, Good I/O evangelist

How to recognize I/O bottlenecks and what to do about them

Rosemary will be sharing industry perspectives on how to recognise I/O bottlenecks and what to do about them. The delicate and often dynamic balance between I/O, CPU and memory can hide some easy wins in terms of improving throughput on-prem and reducing costs in the cloud. Equally, improving I/O is also about reducing the load on shared storage and not just about the incremental improvements of individual applications.

Ellexus Ltd: The I/O Profiling Company

Products: We make system telemetry tools to help you

- improve application performance,
- protect shared storage, and
- manage application dependencies for migration.

Customers include:

Azure diamond tgen AN AFFILIATE OF City of Hope.

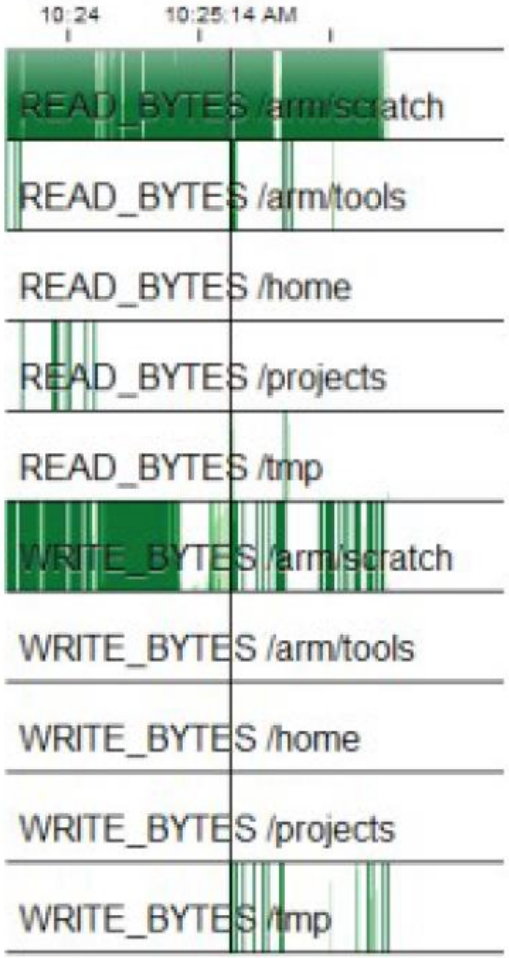
wellcome sanger institute Qualcomm

TACC SYNOPSYS® ANSYS®

Solving the noisy neighbour problem

How we worked with Arm to develop our technology

Example of a rogue job from Arm:



This application is overloading shared storage by putting data in the wrong place.

Temporary data is written to shared storage

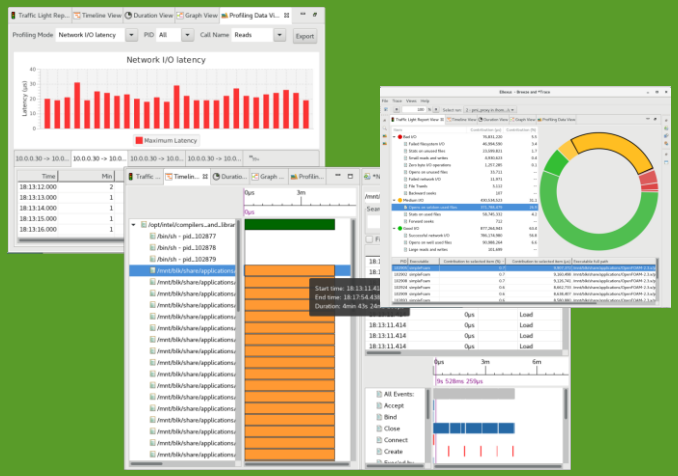
Local storage is unused

Ellexus enterprise products

Take control of the way you access your data

breeze

Detailed I/O profiling
Application discovery



Dependencies
What do I need to include in my container?
How do I migrate this tool chain?

I/O profiling
What resources do I need?

Debug and triage
Why am I not getting the results I expect?

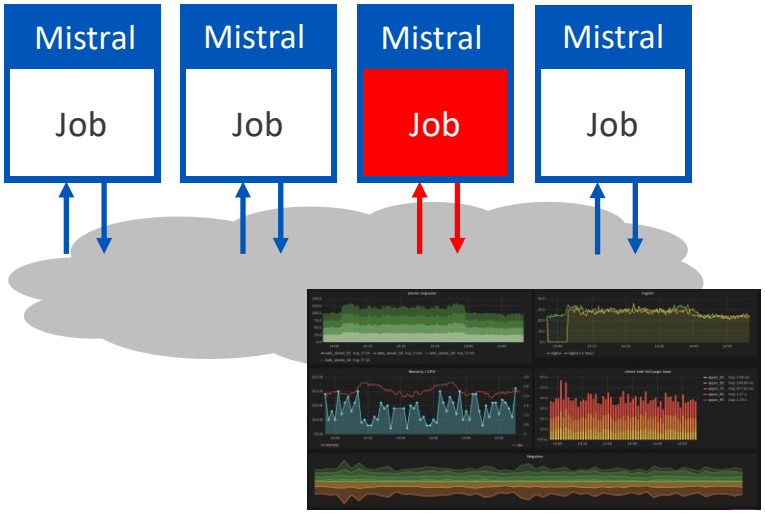
Ellexus enterprise products

Take control of the way you access your data

Live telemetry for on-premises clusters and cloud

Protect storage and find bottlenecks

Cost management and forecasting

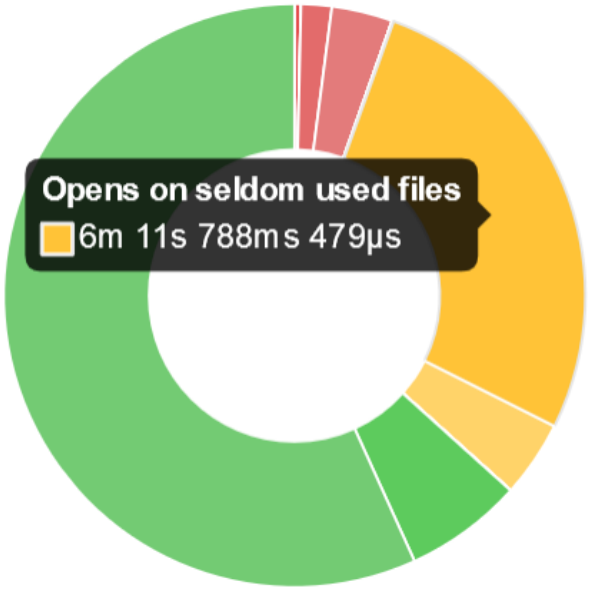


mistral

Live system telemetry:
I/O monitoring in production

Tuning and sizing:

How much time are you wasting doing bad I/O?



- Small reads and writes
- Opens on unused files
- Stats on unused files
- Failed filesystem I/O
- Backward seeks
- File Trawls
- Zero byte I/O operations
- Failed network I/O
- Opens on seldom used files
- Stats on used files
- Forward seeks
- Large reads and writes
- Opens on well used files
- Successful network I/O

breeze

Case study:

Tuning cancer pipelines at the Sanger Institute

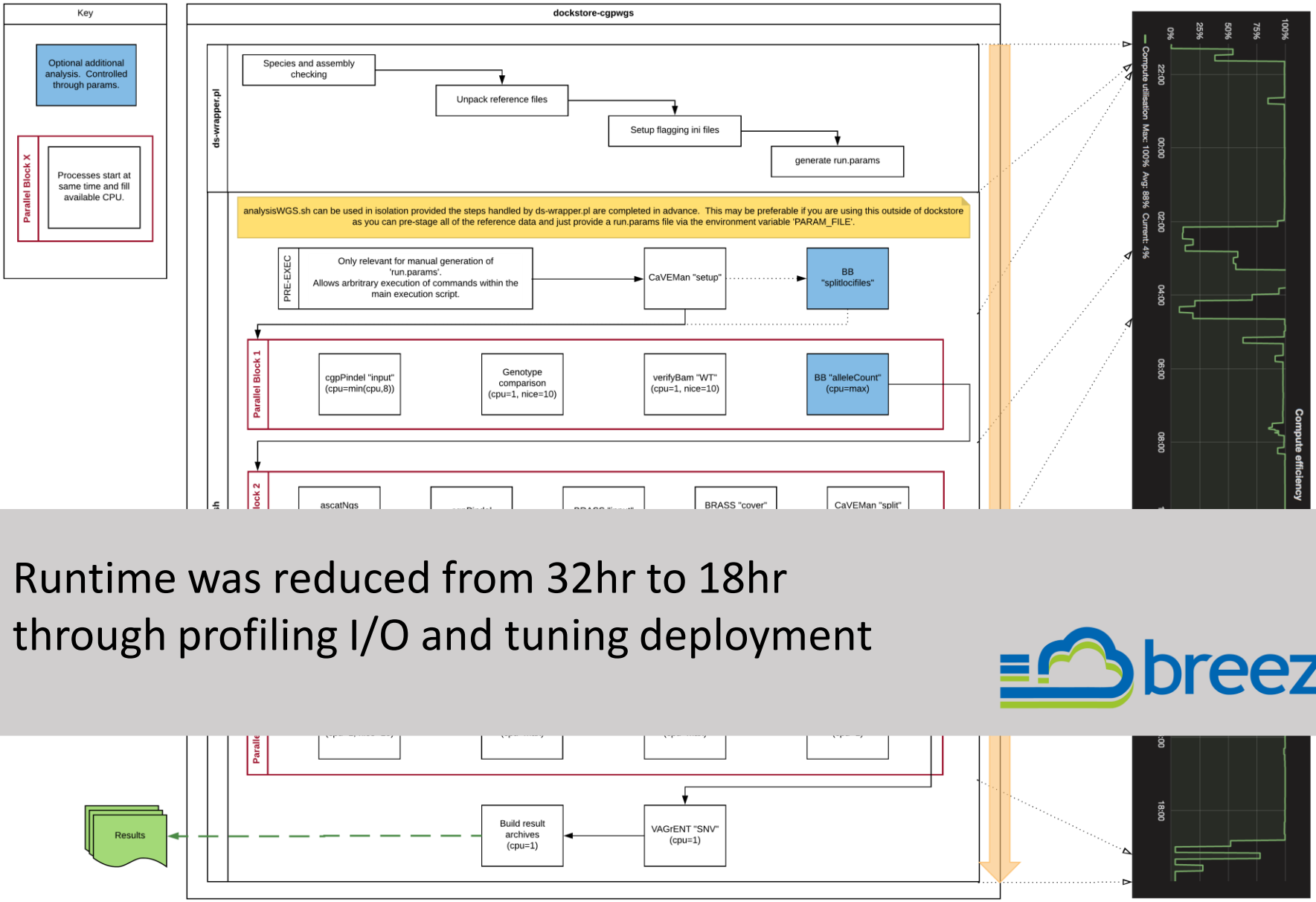
The Pancancer project: 2,000 whole genomes at multiple HPC sites

→ Containerised pipelines for portability

→ I/O tuned with Ellexus tools

→ Storage now needs to be sized correctly

Tuning cancer pipelines at the Sanger Institute



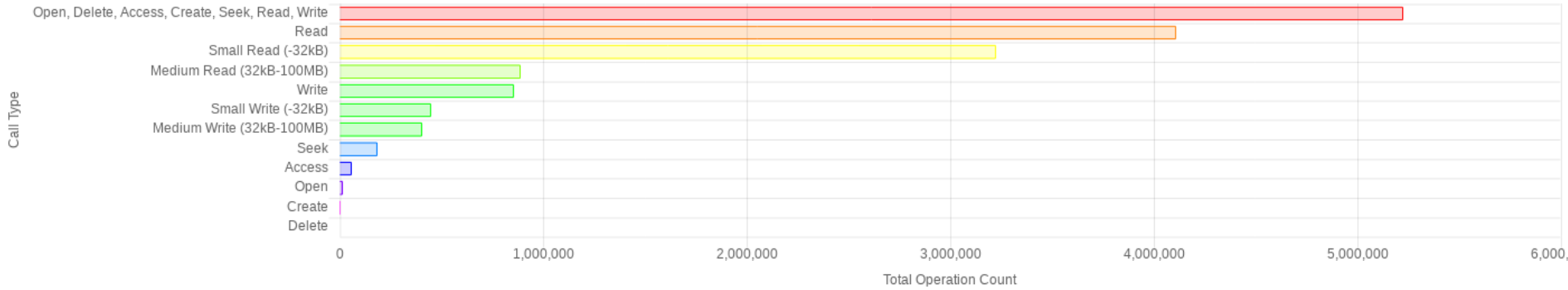
Runtime was reduced from 32hr to 18hr through profiling I/O and tuning deployment

breeze

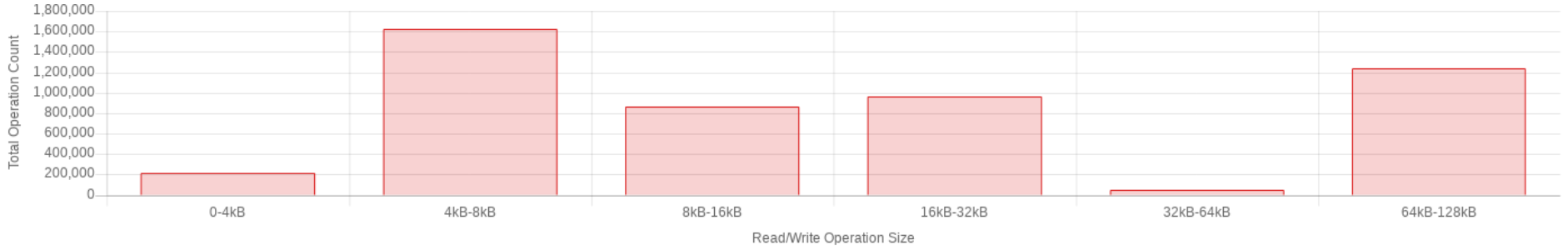
Profiling the cancer pipeline

AWS m5.xlarge 4vCPU 16GB

Number of I/O operations() by type



Size of read and write operations()



Storage comparison

	Time*		Cost per month ($)	
GP2	52m 23s	100%	174.11	100%
Magnetic EBS	1h 01m 44s	118%	174.43	100%
Provisioned 100 IOPS	1h 42m 01s	195%	184.61	106%
Throughput optimised HDD	1h 19m 32s	152%	189.01	109%
150GB NVMe	51m 27s	98%	191.79	110%
Provisioned 500 IOPS	54m 22s	104%	215.01	123%

⇒ The Provisioned IOPS SSDs performed very badly

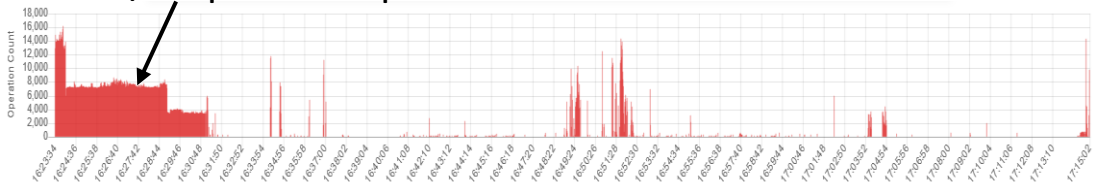
⇒ AWS default option, GP2 is the best

⇒ NVMe was only 2% faster for a 10% price increase

I/O Operations() over time

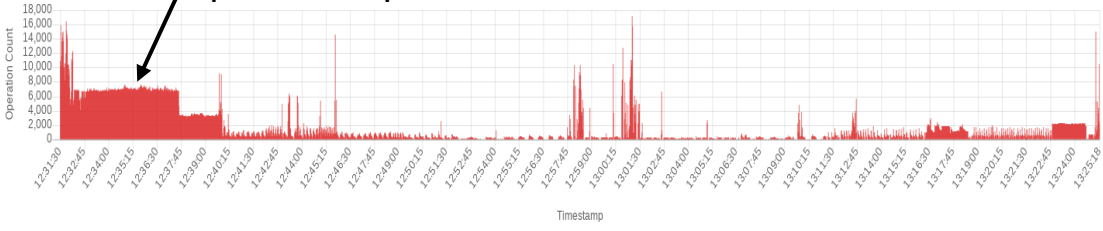
NVMe SSD

~7k I/O operations per second



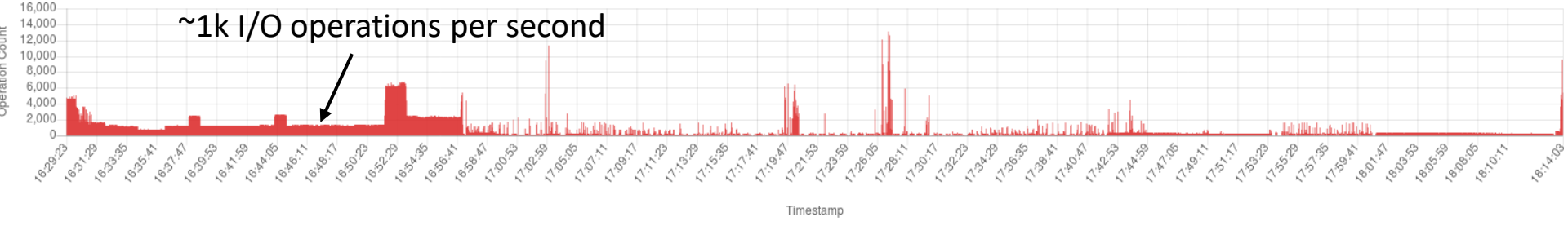
GP2 SSD

~7k I/O operations per second



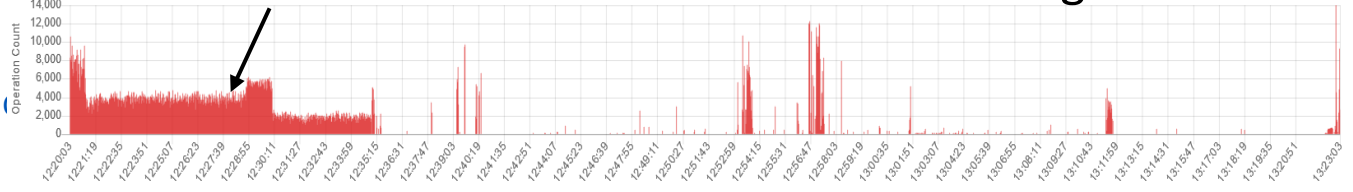
Provisioned 100IOPS SSD

~1k I/O operations per second

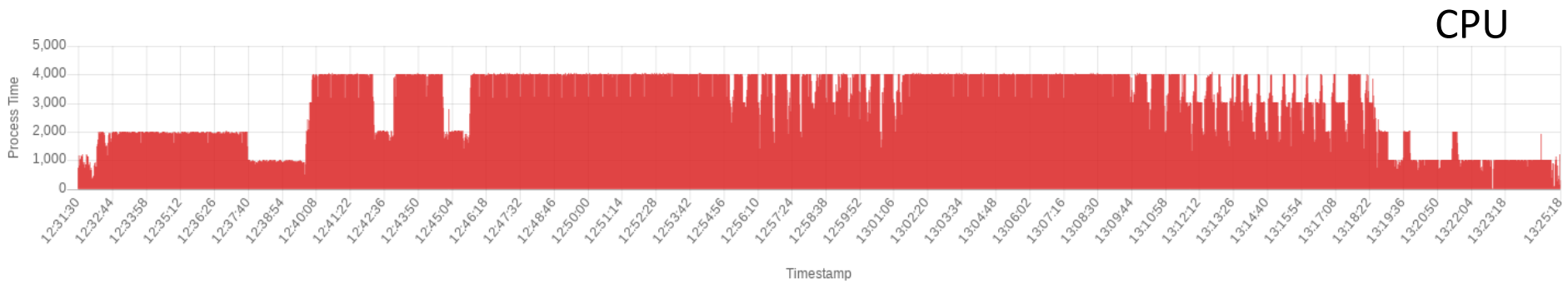


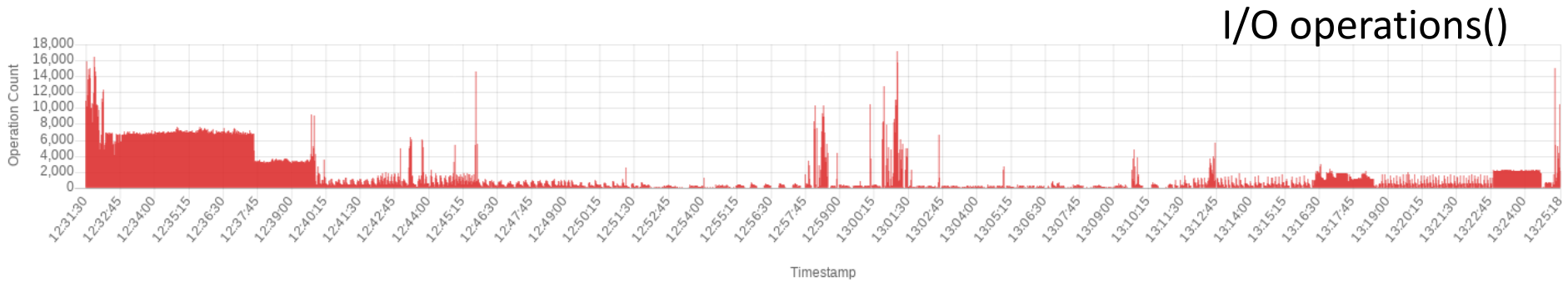
~4k I/O operations per second

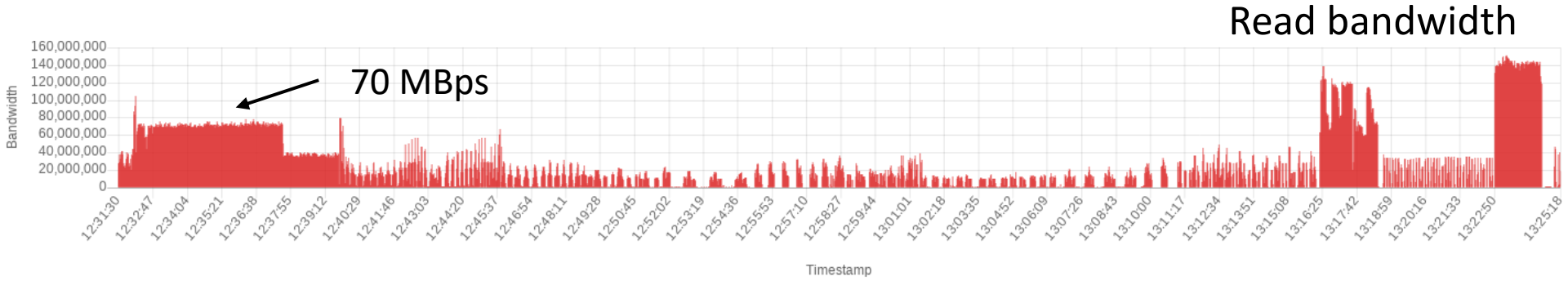
Magnetic HDD



CPU and I/O Profile (on GP2 SSD)





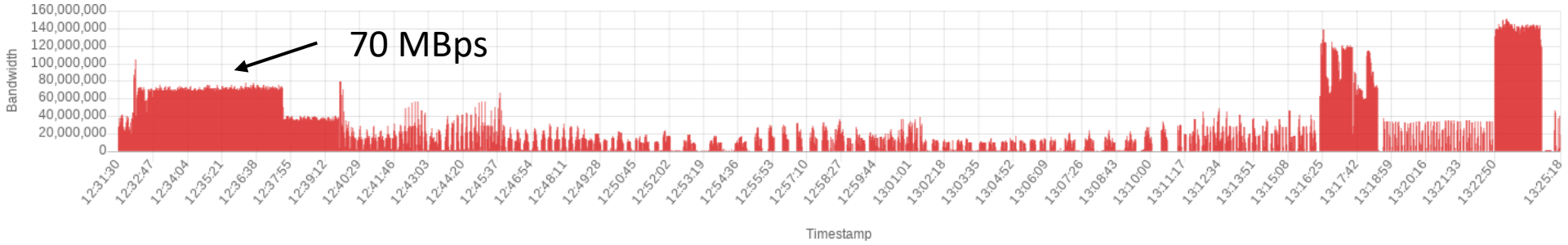


More CPU and less memory: m5.xlarge vs c5.xlarge
(still on GP2 SSD default storage)

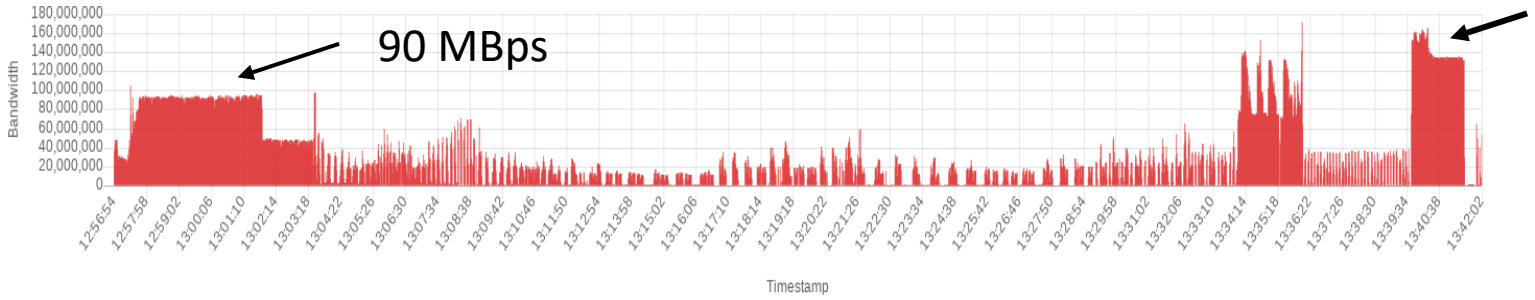
M5.xlarge	**c5.xlarge**
4 vCPU	4 vCPU
16GB	8GB
Runtime: 53min	Runtime: 44min
Cost: $0.21	Cost: $0.16

Read bandwidth: m5.xlarge vs c5.xlarge

Read bandwidth for mx.large 16GB



Read bandwidth for cx.large 8GB



I/O limited by running out of AWS burst credits at the end

How long did this work take?

Sizing the storage and compute correctly took three days

… and we saved 10-40% of cloud costs for the project.

> “Improving run time often doesn't require extensive rewrites. Knowing where to look is key.”
>
> Keiran Raine, Cancer researcher, Sanger Institute

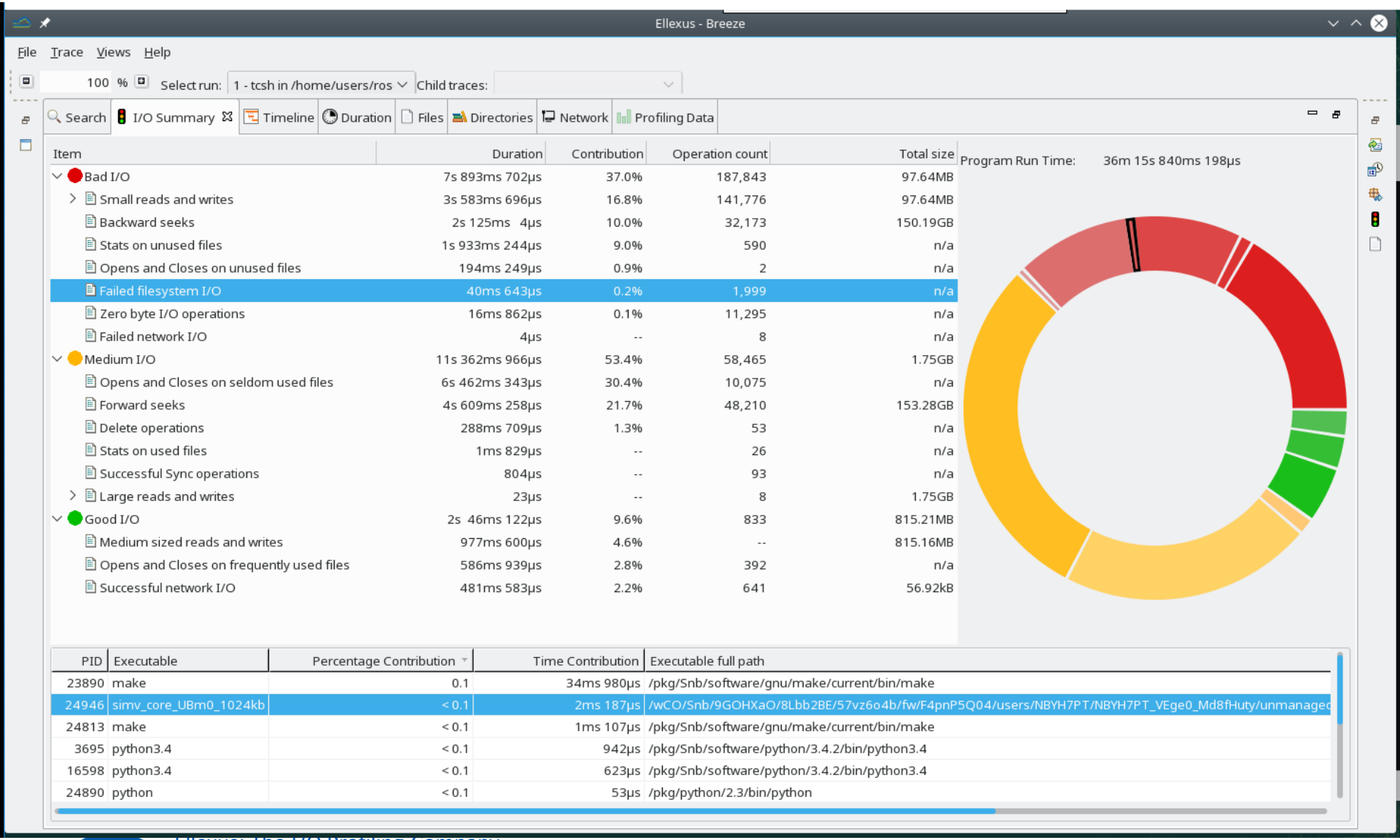
Dependency hygiene flow at Qualcomm

Breeze is used to trace thousands of workflows to automatically identify the mount points, file and network dependencies of each flow for migration.

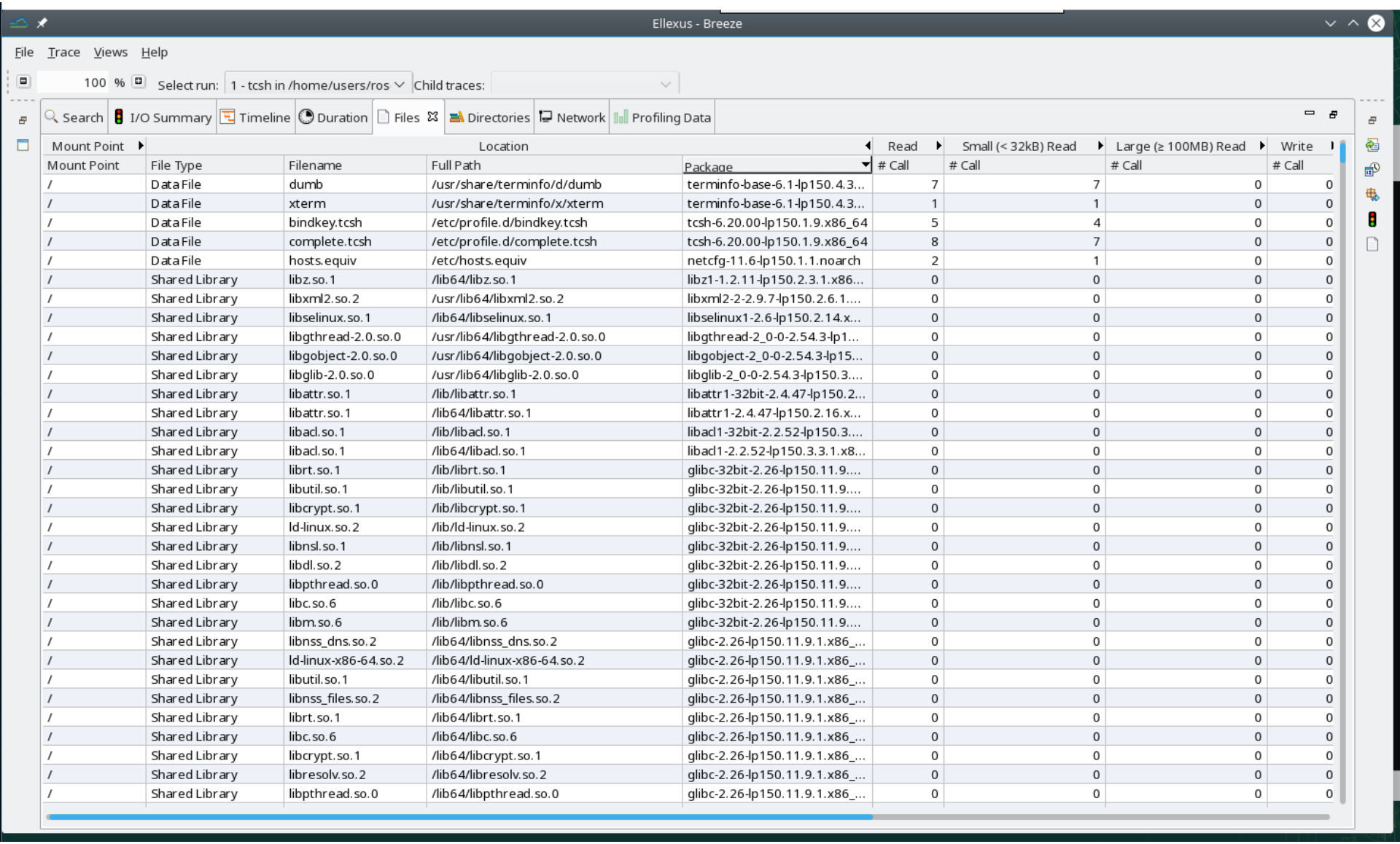
Disclaimer

The following trace was collected at Qualcomm, tracing a Synopsys VCS flow, but all identifiers and data have been modified or removed. No conclusions can be drawn from the following screenshots about the IT infrastructure, tools or usage at Qualcomm or Synopsys.

Dependency hygiene flow at Qualcomm



File dependencies, mounts points, packages for migration and containerization



Mount Point	Location			Read	Small (< 32kB) Read	Large (≥ 100MB) Read	Write
Mount Point	File Type	Filename	Full Path	Package	# Call	# Call	# Call	# Call
/	Data File	dumb	/usr/share/terminfo/d/dumb	terminfo-base-6.1-lp150.4.3...	7	7	0	0
/	Data File	xterm	/usr/share/terminfo/x/xterm	terminfo-base-6.1-lp150.4.3...	1	1	0	0
/	Data File	bindkey.tcsh	/etc/profile.d/bindkey.tcsh	tcsh-6.20.00-lp150.1.9.x86_64	5	4	0	0
/	Data File	complete.tcsh	/etc/profile.d/complete.tcsh	tcsh-6.20.00-lp150.1.9.x86_64	8	7	0	0
/	Data File	hosts.equiv	/etc/hosts.equiv	netcfg-11.6-lp150.1.1.noarch	2	1	0	0
/	Shared Library	libz.so.1	/lib64/libz.so.1	libz1-1.2.11-lp150.2.3.1.x86...	0	0	0	0
/	Shared Library	libxml2.so.2	/usr/lib64/libxml2.so.2	libxml2-2-2.9.7-lp150.2.6.1....	0	0	0	0
/	Shared Library	libselinux.so.1	/lib64/libselinux.so.1	libselinux1-2.6-lp150.2.14.x...	0	0	0	0
/	Shared Library	libgthread-2.0.so.0	/usr/lib64/libgthread-2.0.so.0	libgthread-2_0-0-2.54.3-lp1...	0	0	0	0
/	Shared Library	libgobject-2.0.so.0	/usr/lib64/libgobject-2.0.so.0	libgobject-2_0-0-2.54.3-lp15...	0	0	0	0
/	Shared Library	libglib-2.0.so.0	/usr/lib64/libglib-2.0.so.0	libglib-2_0-0-2.54.3-lp150.3....	0	0	0	0
/	Shared Library	libattr.so.1	/lib/libattr.so.1	libattr1-32bit-2.4.47-lp150.2...	0	0	0	0
/	Shared Library	libattr.so.1	/lib64/libattr.so.1	libattr1-2.4.47-lp150.2.16.x...	0	0	0	0
/	Shared Library	libacl.so.1	/lib/libacl.so.1	libacl1-32bit-2.2.52-lp150.3....	0	0	0	0
/	Shared Library	libacl.so.1	/lib64/libacl.so.1	libacl1-2.2.52-lp150.3.3.1.x8...	0	0	0	0
/	Shared Library	librt.so.1	/lib/librt.so.1	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	libutil.so.1	/lib/libutil.so.1	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	libcrypt.so.1	/lib/libcrypt.so.1	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	ld-linux.so.2	/lib/ld-linux.so.2	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	libnsl.so.1	/lib/libnsl.so.1	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	libdl.so.2	/lib/libdl.so.2	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	libpthread.so.0	/lib/libpthread.so.0	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	libc.so.6	/lib/libc.so.6	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	libm.so.6	/lib/libm.so.6	glibc-32bit-2.26-lp150.11.9....	0	0	0	0
/	Shared Library	libnss_dns.so.2	/lib64/libnss_dns.so.2	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0
/	Shared Library	ld-linux-x86-64.so.2	/lib64/ld-linux-x86-64.so.2	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0
/	Shared Library	libutil.so.1	/lib64/libutil.so.1	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0
/	Shared Library	libnss_files.so.2	/lib64/libnss_files.so.2	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0
/	Shared Library	librt.so.1	/lib64/librt.so.1	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0
/	Shared Library	libc.so.6	/lib64/libc.so.6	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0
/	Shared Library	libcrypt.so.1	/lib64/libcrypt.so.1	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0
/	Shared Library	libresolv.so.2	/lib64/libresolv.so.2	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0
/	Shared Library	libpthread.so.0	/lib64/libpthread.so.0	glibc-2.26-lp150.11.9.1.x86_...	0	0	0	0

Profiling EDA tools: Xilinx Vivado

Ellexus - Breeze

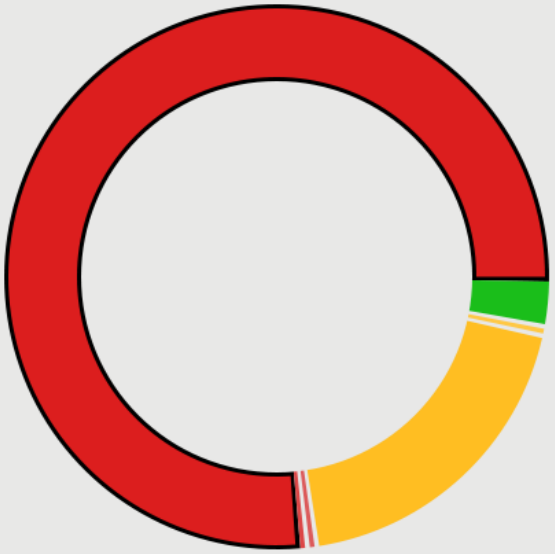
File Trace Views Help

100 % Select run: 3 - run_synth.sh in /ell Child traces:

Search | I/O Summary | Timeline | Duration | Files | Directories | Network | Profiling Data

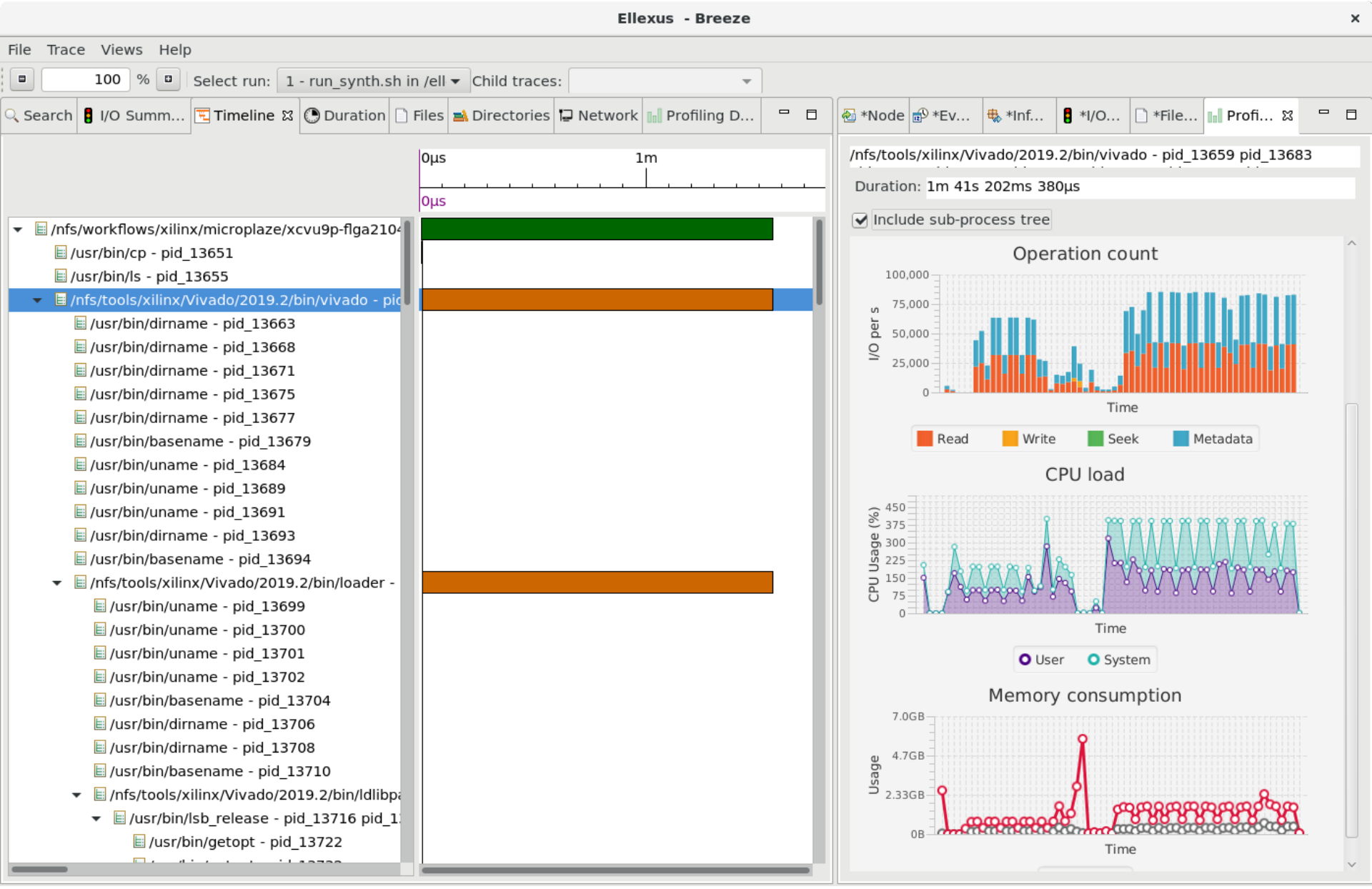
Item	Duration	Contribution	Operation count	Total size
Bad I/O	37s 120ms 358µs	77.8%	1,522,129	1.42GB
Small reads and writes	36s 669ms 486µs	77.0%	1,366,287	1.42GB
Small reads	36s 653ms 936µs	77.0%	1,358,099	1.40GB
Small writes	15ms 550µs	--	8,188	15.02MB
Stats on unused files	224ms 922µs	0.4%	66,292	n/a
Failed filesystem I/O	201ms 748µs	0.4%	70,668	n/a
Zero byte I/O operations	13ms 943µs	--	16,245	n/a
Opens and Closes on unused files	10ms 18µs	--	2,205	n/a
Backward seeks	241µs	--	432	420.68MB
Failed network I/O	--	--	--	n/a
Medium I/O	9s 342ms 895µs	19.6%	1,317,594	--
Opens and Closes on seldom used files	9s 214ms 719µs	19.4%	1,276,942	n/a
Stats on used files	122ms 533µs	0.2%	38,378	n/a
Delete operations	4ms 849µs	--	517	n/a
Successful Sync operations	594µs	--	1,410	n/a
Forward seeks	200µs	--	347	420.56MB
Large reads and writes	--	--	--	--
Good I/O	1s 247ms 780µs	2.6%	1,893	451.77MB
Medium sized reads and writes	1s 230ms 599µs	2.6%	994	451.76MB
Opens and Closes on frequently used files	16ms 895µs	--	857	n/a
Successful network I/O	286µs	--	42	13.52kB

Program Run Time: 1m 40s 567ms 974µs



PID	Executable	Percentage Contribution	Time Contribution	Executable full path
14087	vivado	17.5	14s 803ms 884µs	/nfs/tools/xilinx/Vivado/2019.2/bin/unwrapped/lnx64.o/vivado
14090	vivado	17.3	14s 584ms 890µs	/nfs/tools/xilinx/Vivado/2019.2/bin/unwrapped/lnx64.o/vivado
13765	vivado	8.3	6s 977ms 654µs	/nfs/tools/xilinx/Vivado/2019.2/bin/unwrapped/lnx64.o/vivado
13775	srcscanner	< 0.1	61ms 815µs	/nfs/tools/xilinx/Vivado/2019.2/bin/unwrapped/lnx64.o/srcscanner
13801	srcscanner	< 0.1	61ms 212µs	/nfs/tools/xilinx/Vivado/2019.2/bin/unwrapped/lnx64.o/srcscanner
13795	srcscanner	< 0.1	61ms 25µs	/nfs/tools/xilinx/Vivado/2019.2/bin/unwrapped/lnx64.o/srcscanner
13780	srcscanner	< 0.1	61ms 24µs	/nfs/tools/xilinx/Vivado/2019.2/bin/unwrapped/lnx64.o/srcscanner
13651	cp	< 0.1	34ms 339µs	/usr/bin/cp
13882	vrs	< 0.1	22ms 138µs	/nfs/tools/xilinx/Vivado/2019.2/bin/unwrapped/lnx64.o/vrs

Profiling EDA tools: Xilinx Vivado

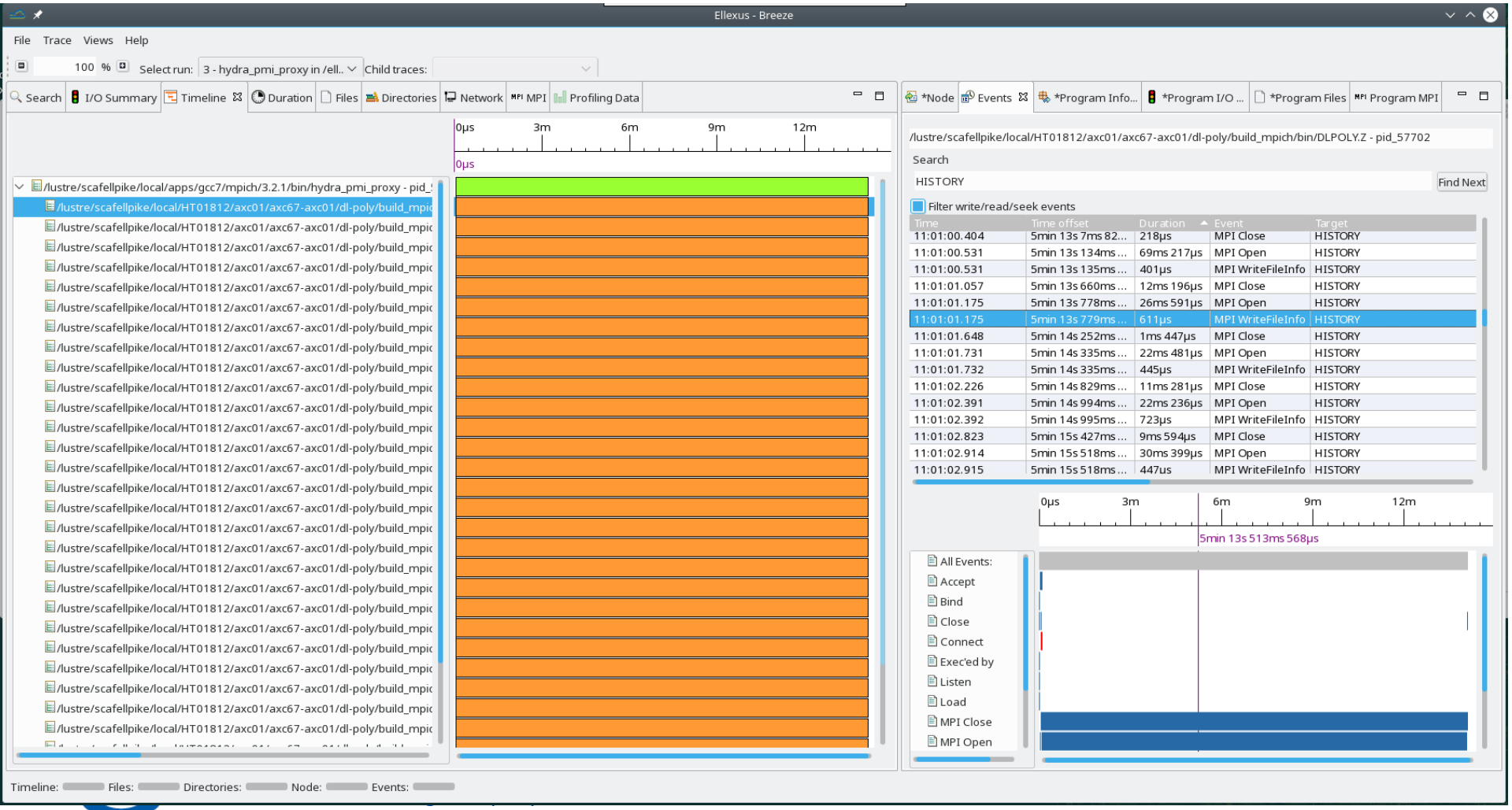


I/O profiling to improve DL_POLY with the Hartree Centre

DL_POLY is a general purpose classical molecular dynamics tool that uses MPI I/O.

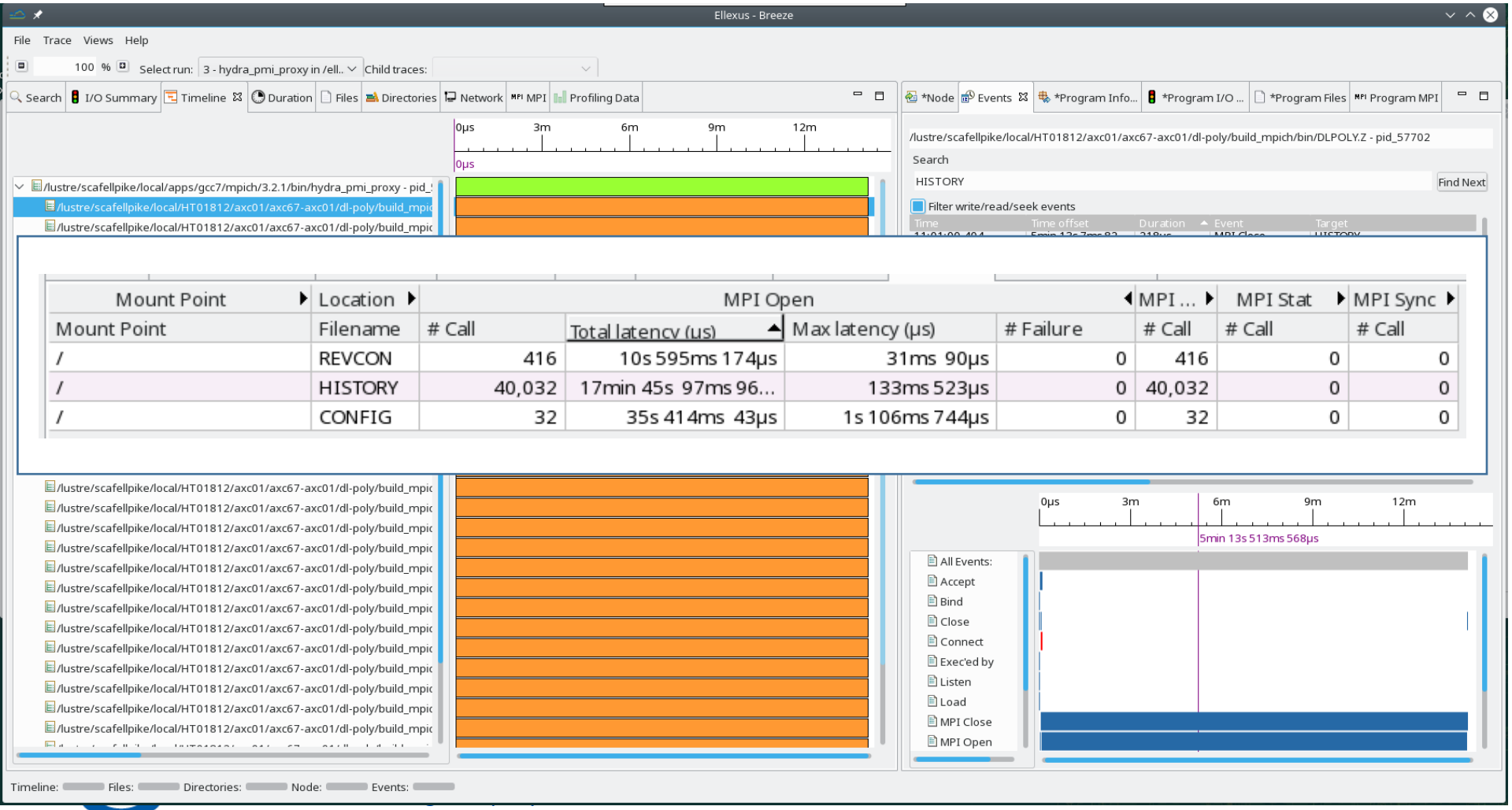
I/O profiling to improve DL_POLY with the Hartree Centre

DL_POLY is a general purpose classical molecular dynamics tool that uses MPI I/O.
It was opening the HISTORY file from ever rank, but only writing from one.

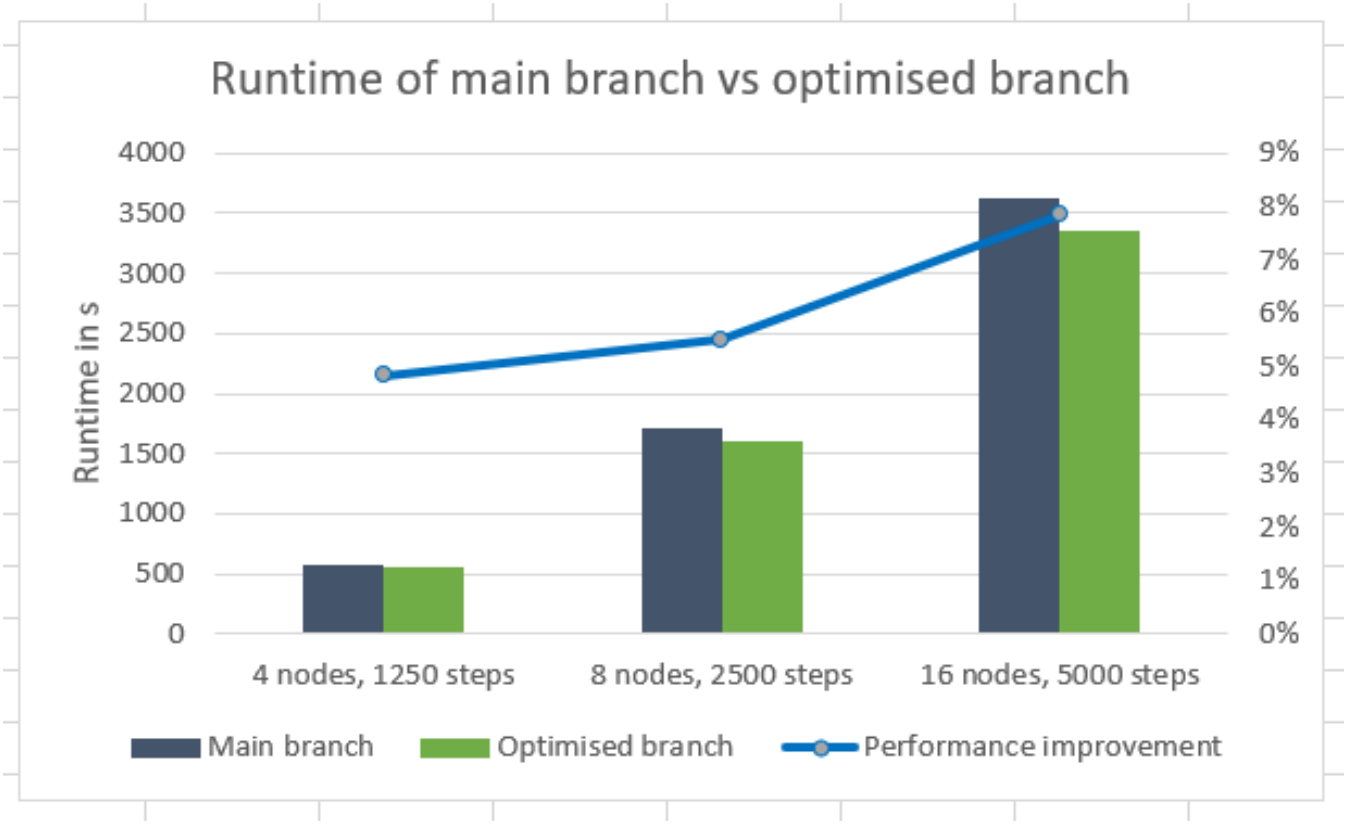


I/O profiling to improve DL_POLY with the Hartree Centre

DL_POLY is a general purpose classical molecular dynamics tool that uses MPI I/O.
It was opening the HISTORY file from ever rank, but only writing from one.



I/O profiling to improve DL_POLY with the Hartree Centre



Removing unnecessary opens gave significant performance improvements

I/O profiling and what to do with the results

Profile in production

Optimization

Steering

- Data location, Scheduling, Filesystem, Burst buffers

Ellexus Ltd: The I/O Profiling Company

Dr Rosemary Francis, CEO, Good I/O evangelist

Thanks for listening

The I/O Profiling Company - Protect. Balance. Optimise.

www.ellexus.com