TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Dr. Andreas Knüpfer
Zentrum für Informationsdienste und Hochleistungsrechnen (ZIH)

HPC und Daten: Nutzungskonzepte und Einsichten

10. HPC-Statuskonferenz der Gauß-Allianz

1. Oktober 2020

ZIH
Center for Information Services & High Performance Computing

Datenintensives HPC am ZIH und im ScaDS.AI

Datenintensives HPC am ZIH

Kontinuierlicher Schwerpunkt in den HPC-Hardware-Konzepten

- „Hochleistungsrechner-Speicher-Komplex“ (HRSK) 2005
- HRSK-II 2013/2015
- HPC Data Analytics Installation 2018-2020
- Fortgesetzt im NHR-Antrag (in Begutachtung)

ScaDS.AI Dresden/Leipzig

- Nationales Kompetenzzentrum für Big Data ScaDS Dresden/Leipzig seit 2014
- Seit 2019 ScaDS.AI Dresden/Leipzig (“Center for Scalable Data Analytics and Artificial Intelligence”)
- Von TU Dresden und Universität Leipzig gemeinsam beantragt und betrieben
- Forschungs- und Service-Auftrag
- Finanziert von BMBF und dem Land Sachsen

Warum ist I/O eine besondere Performance-Kategorie?

Was ist wichtig im High Performance Computing?

- Flop/s sind wichtig
- Hauptspeicher-Größe und Bandbreiten sind wichtig
- I/O ist wichtig bei großen Datenmengen

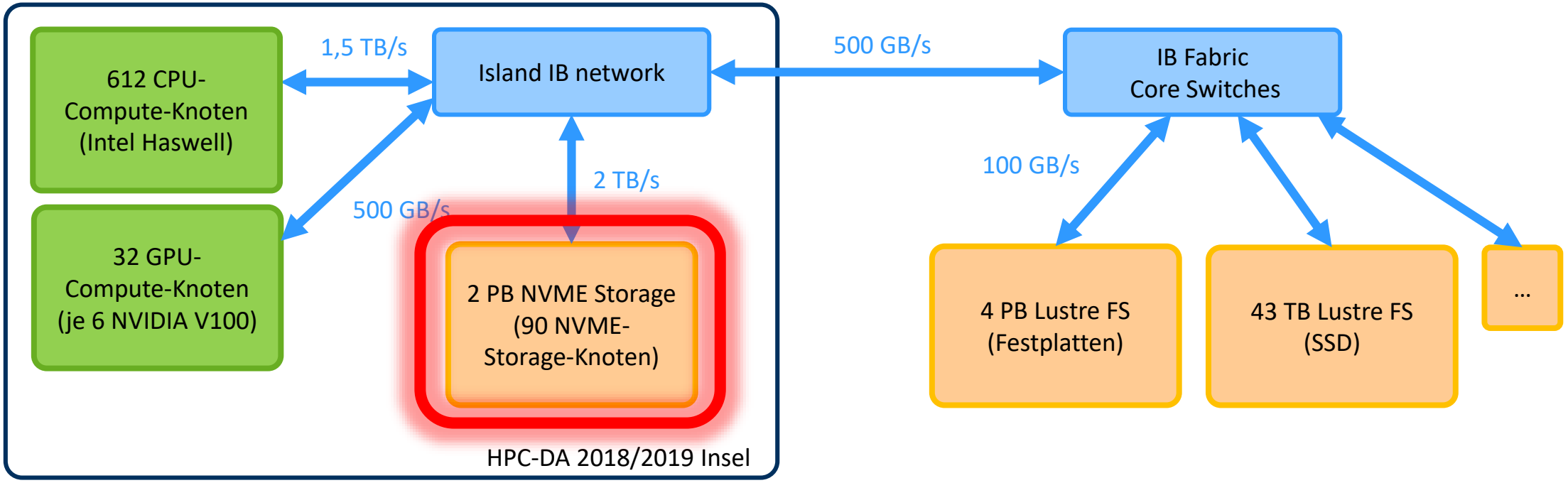
Dafür kaufen wir einfach die richtige Hardware, oder?

… leider nicht ganz so einfach für I/O-Performance

Ausschnitt aus der aktuellen HPC-Installation der TU Dresden

— HPC-DA Cluster-Insel mit Compute-Knoten und dedizierten Storage-Knoten

— Separates Infiniband-Netz innerhalb der Insel mit hohen Bandbreiten



NVME Storage-Knoten

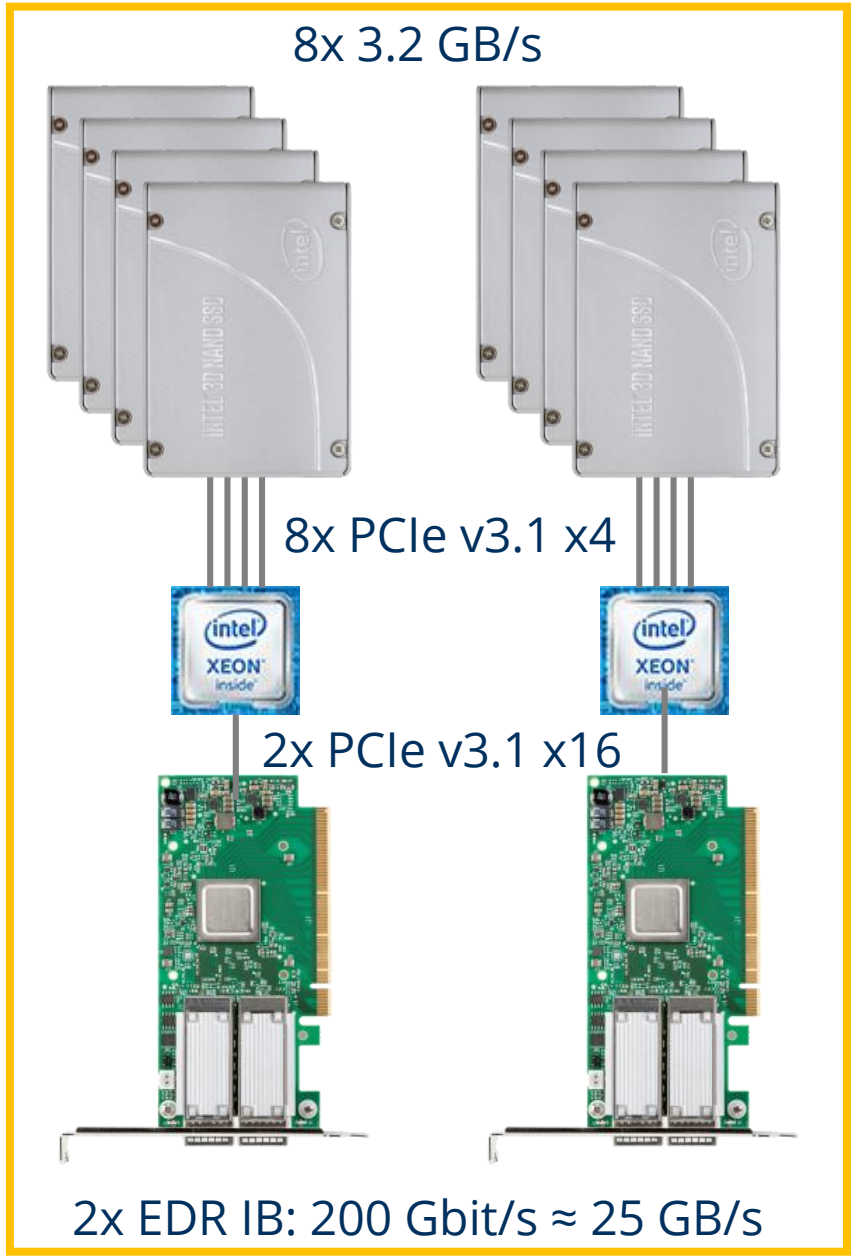
- 90 NVME Storage-Knoten
- Je 2 Intel Xeon E5-2620 v4 CPUS (16 Kerne, 2.10GHz), 64 GB RAM

- 8x Intel NVMe Datacenter SSD P4610, (PCIe 3.1 x4 3DNAND ME 2.5" U.2)
- 3,2 GB/s pro Karte, 25.6 GB/s pro Knoten
- 2 Infiniband EDR links, Mellanox MT27800, ConnectX-5, PCIe x16, je 100 Gbit/s

- BeeGFS als paralleles Dateisystem

90 x
NVME
Storage-Knoten

Insgesamt
2 TB/s
Bandbreite



Gemeinsam genutzte vs. separate Ressourcen für I/O

Gemeinsam genutzte Storage-Ressourcen

- Parallele Dateisysteme
  - Üblicherweise ein oder wenige pro HPC-Zentrum
  - … tlw. mit verschiedenen Charakteristiken
  - Paralleler Zugriff von allen Compute-Knoten aus, durch alle Arbeitsgruppen und Anwendungen
- Storage-Hardware
  - Daten-Speicher (OTS)
  - Metadaten-Management und Speicher (MDS)
- Netzwerk als Transportweg

Üblicherweise stark schwankende I/O-Phasen

→ Gegenseitige Interferenzen

→ Nicht innerhalb einer Anwendung lösbar

Separate Ressourcen

- Eigenes paralleles Dateisystem pro Arbeitsgruppe
  - Temporär für ein HPC-Projekt
  - Für ausgewählte Arbeitsgruppen
  - Paralleler Zugriff auf ein Dateisystem, gemeinsame Nutzung durch alle Jobs der Gruppe
- Storage-Hardware
  - Eigene OTS → keine Bandbreiten-Konkurrenz
  - Eigene MDS →keine Metadaten-Interferenz
- Netzwerk bleibt gemeinsamer Transportweg

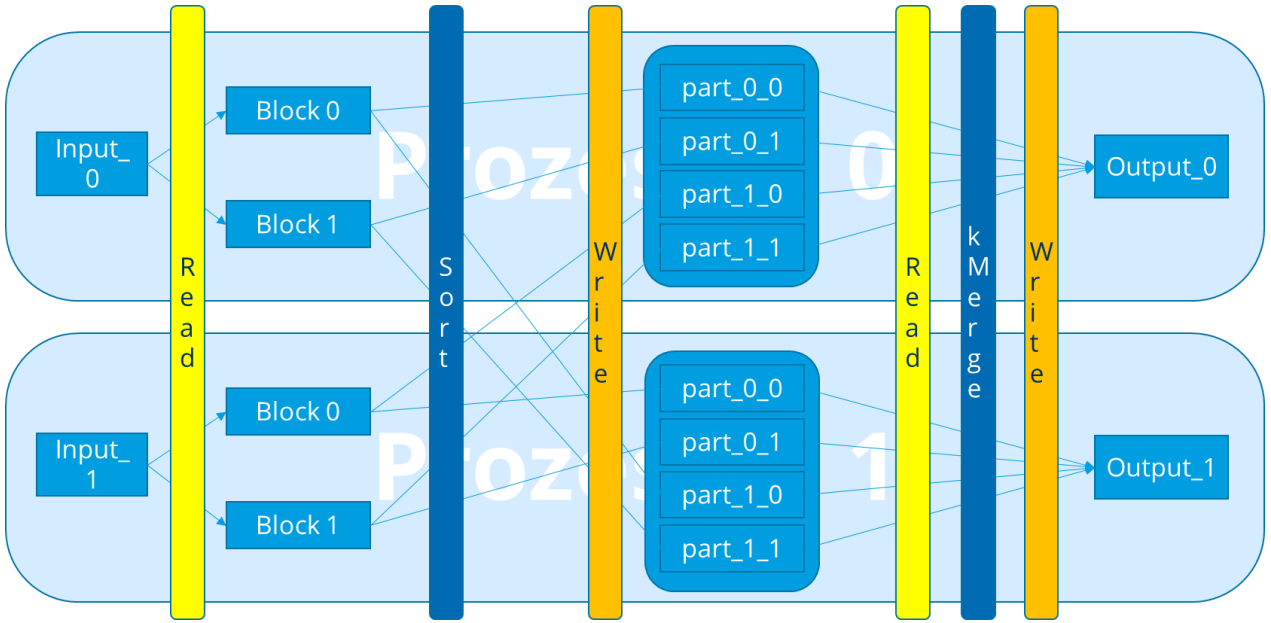
Weniger Performance-Beeinflussung

von anderen und auf andere

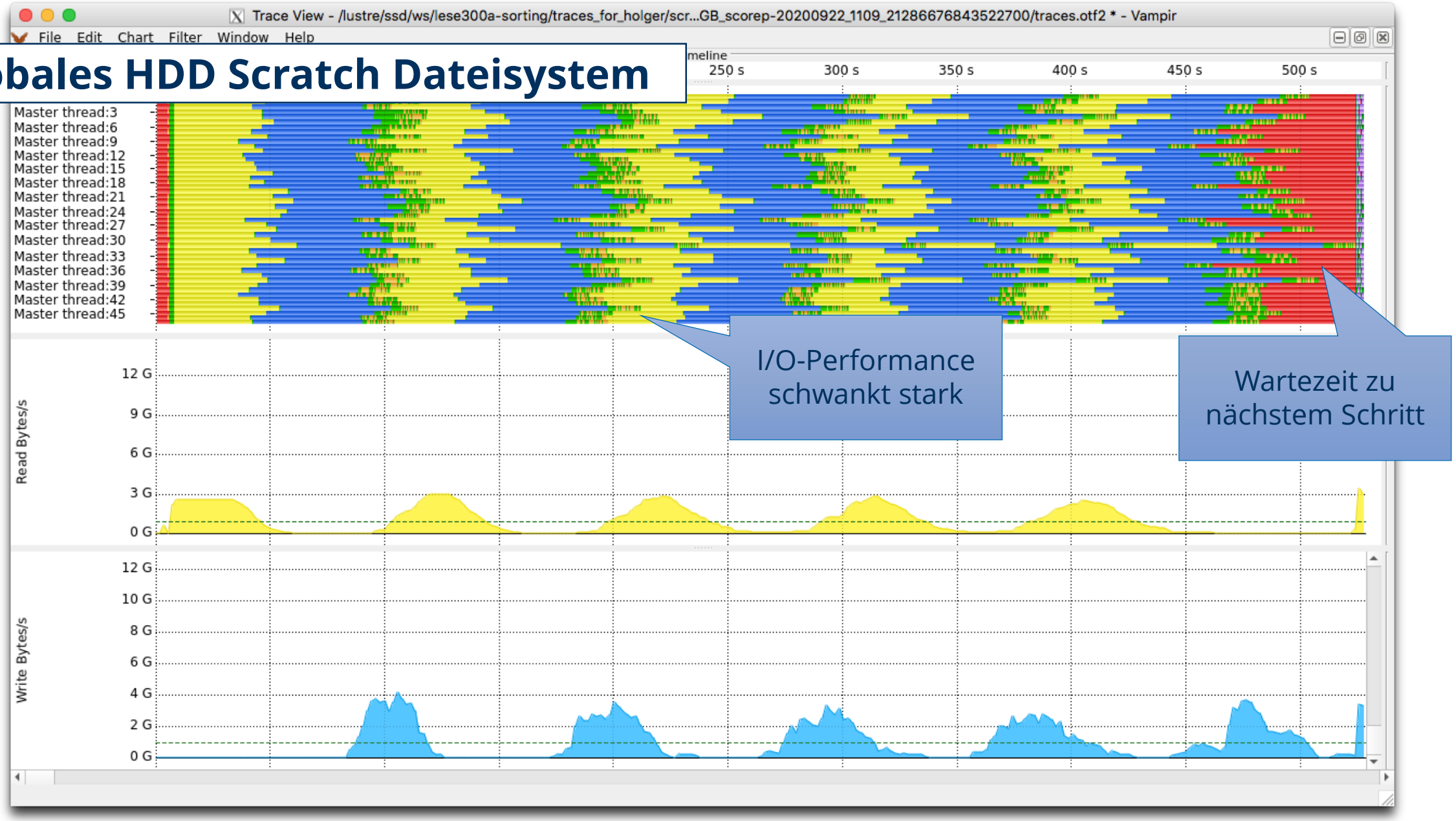
Extra Verwaltungsaufwand

Synthetische Evaluierung: Verteiltes, I/O-basiertes Sortieren

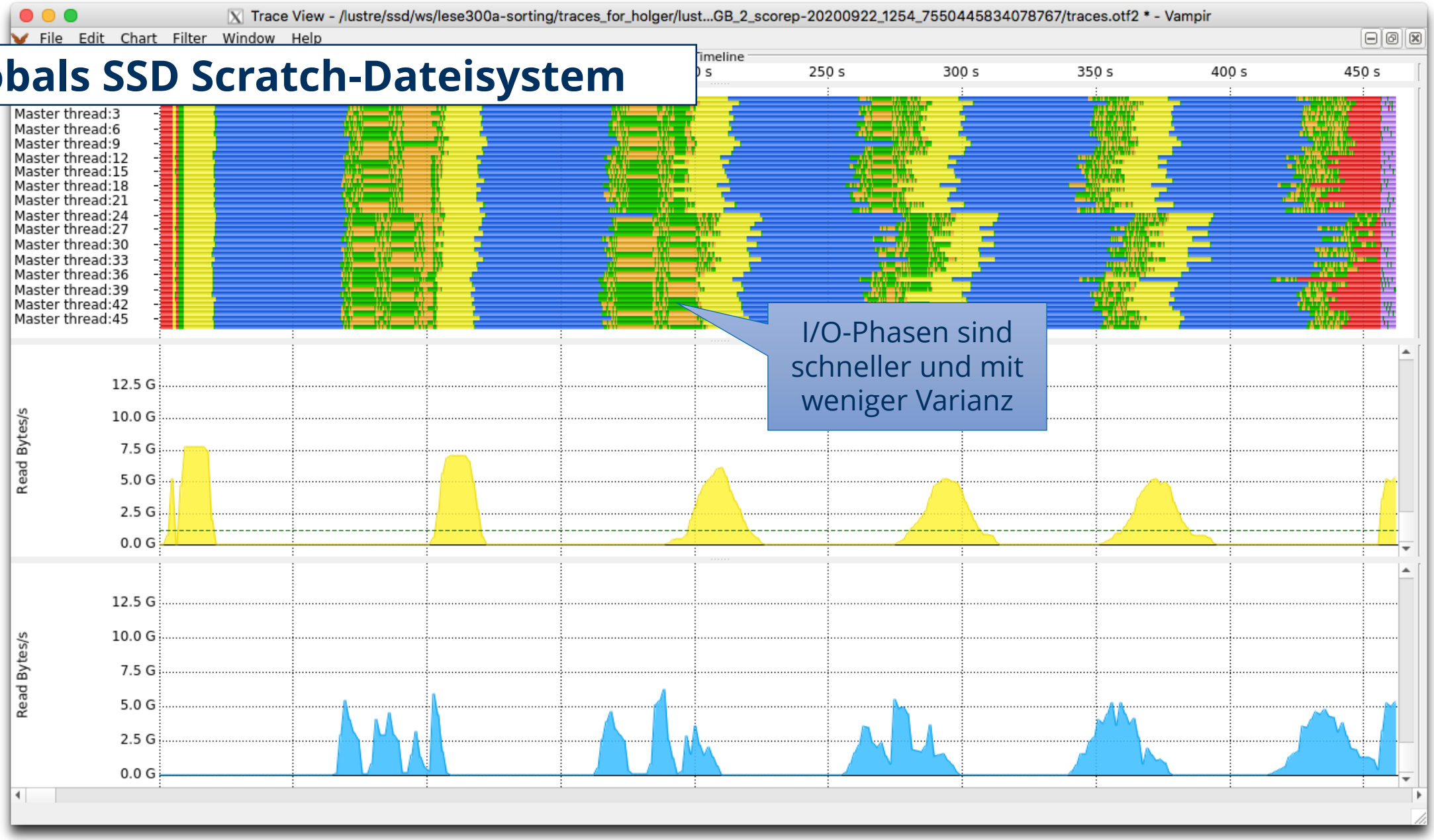
- Inspiriert durch die GraySort-Challenge
- Rapid-Prototyping-Version in Python
  - Effizienz der Berechnung egal
- Optimierte C++ Umsetzung
  - Effizienteres, mehrstufiges I/O-Muster
  - nicht mehr durch RAM limitiert
- MPI nur zur Ablaufkontrolle
- Daten werden via Dateisystem sortiert
- Jeder Prozess liest eine Eingabedatei und erzeugt eine Ergebnis-Datei
- Keine gleichzeitigen Zugriffe mehrerer Prozesse auf Dateien



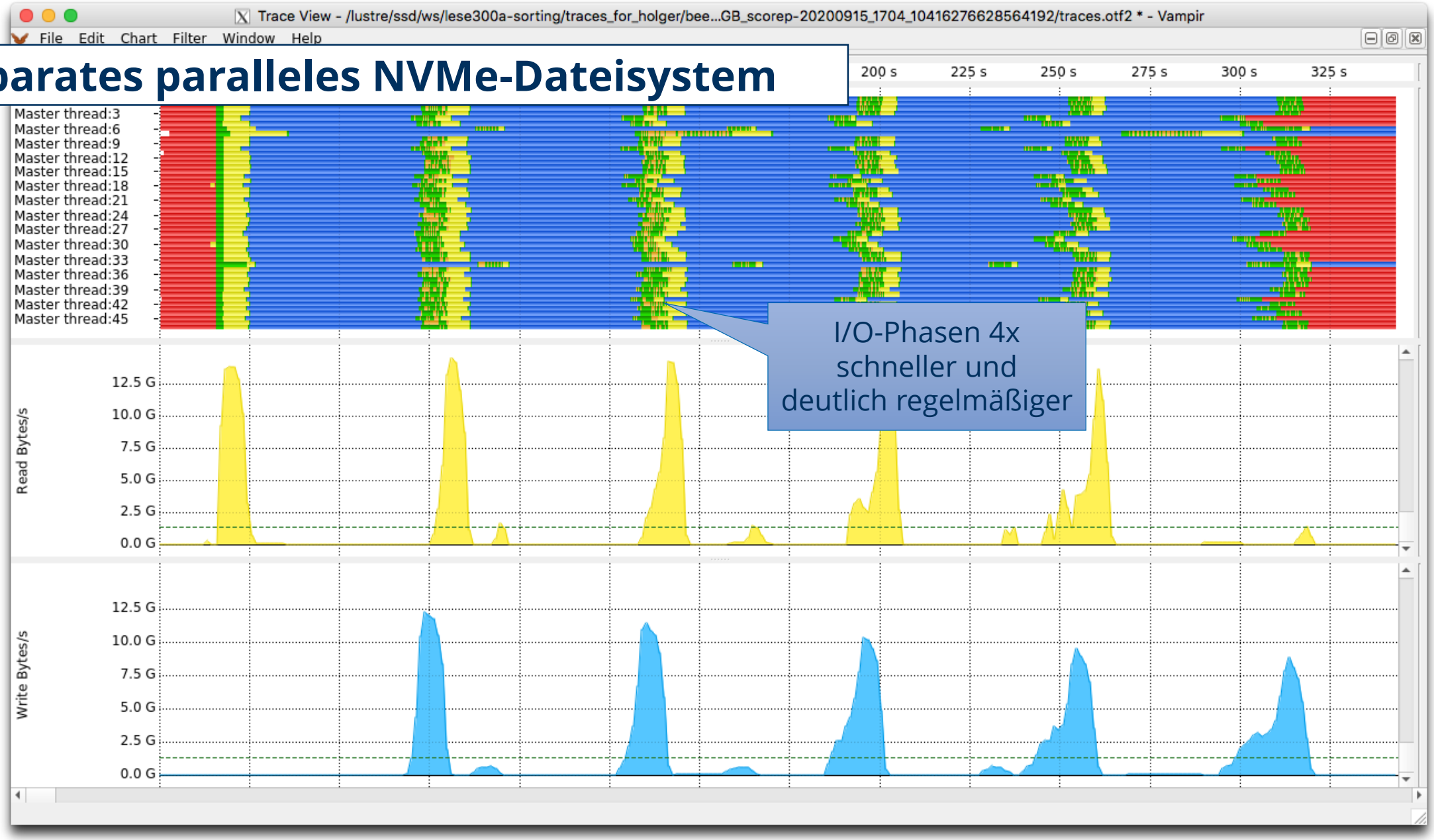
Globales HDD Scratch Dateisystem



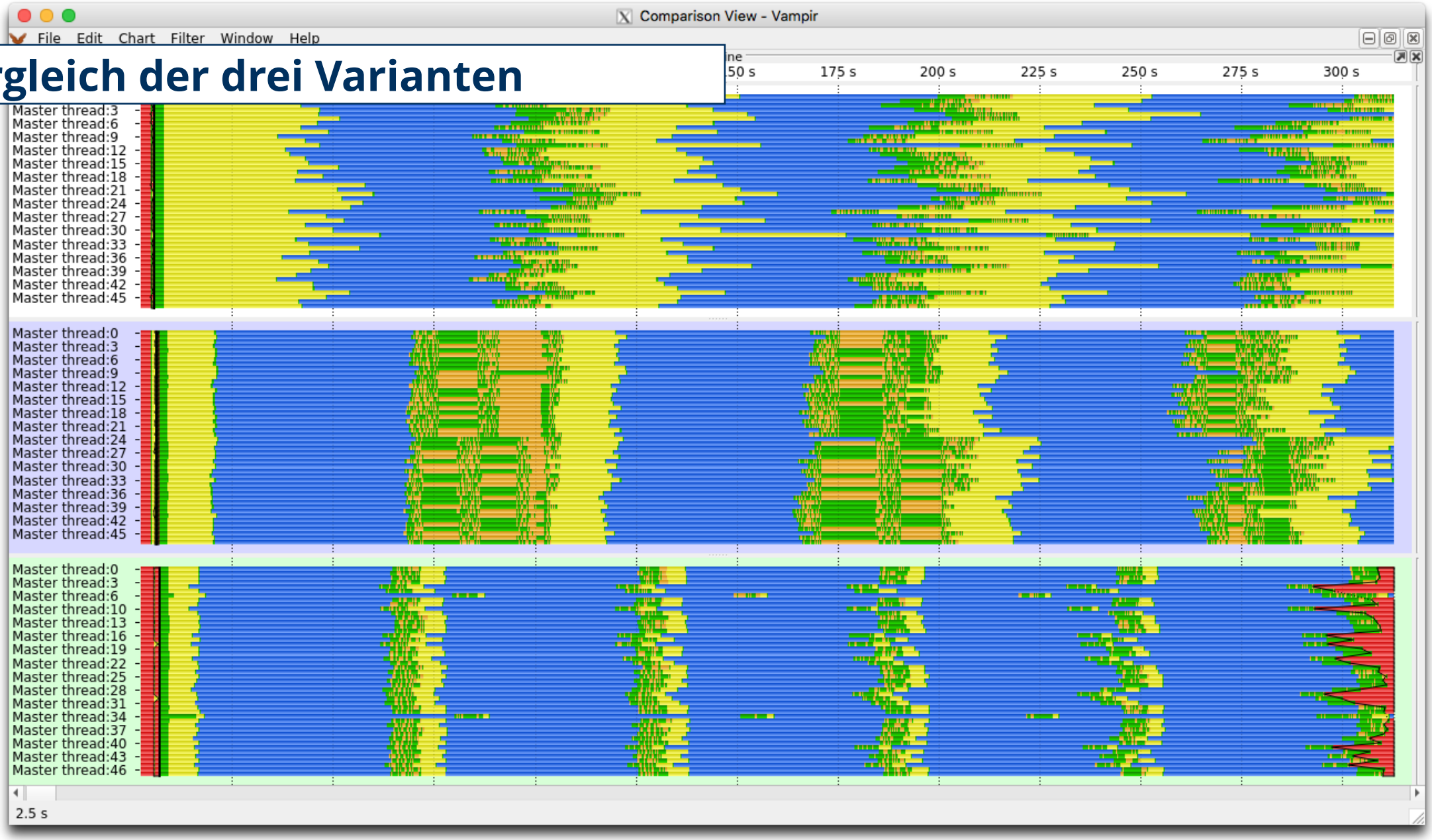
Globals SSD Scratch-Dateisystem



Separates paralleles NVMe-Dateisystem

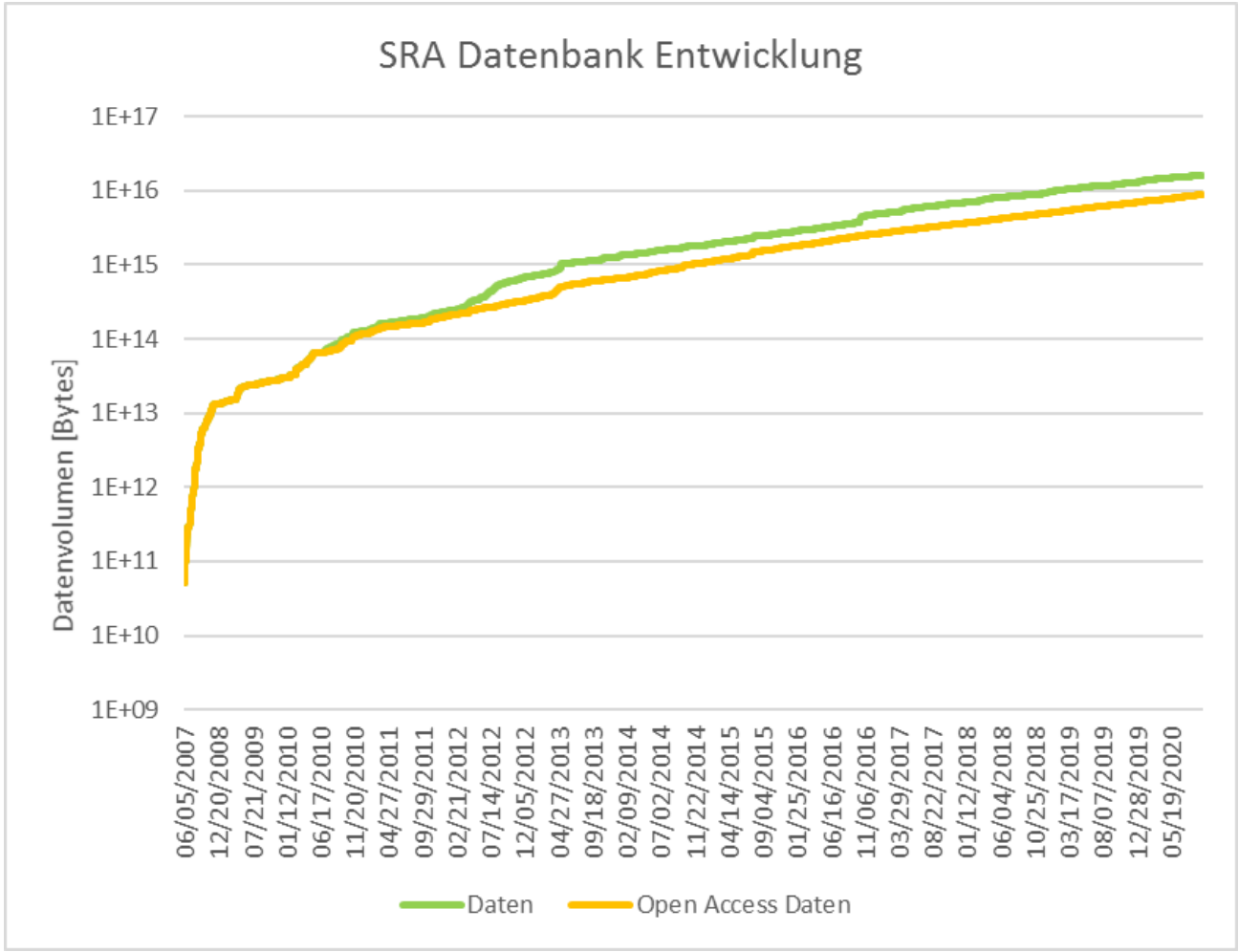


Vergleich der drei Varianten



Anwendungsbeispiel „Discovering Unknown Viruses from Sequencing Data“

- Umfangreiche Genome-Sequenz-Daten aus dem Sequence Read Archive (SRA) https://www.ncbi.nlm.nih.gov/sra/
- 6,343,973 Datensätze, davon ca. 1,5 Million menschlichen Ursprungs (Stand Sept. 2020), > 15 PB (Sept. 2020)
- Zweistufige Analyse zum Aufspüren von Virus-DNA in Genome-Datensätzen
- Chris Lauber (Twincore, Hannover) und Stefan Seitz (DKFZ, Heidelberg)

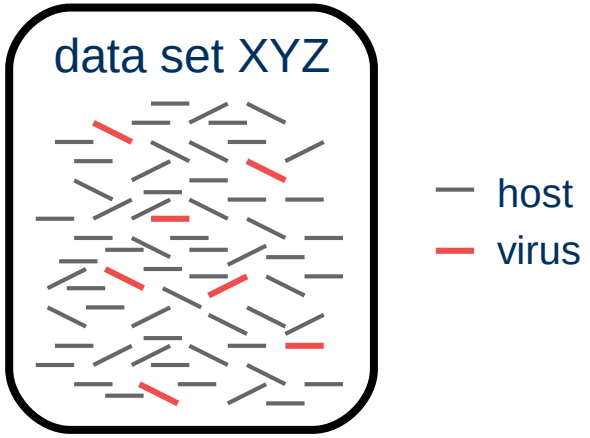


https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi /
https://trace.ncbi.nlm.nih.gov/Traces/sra/sra_stat.cgi

Anwendungen auf dedizierten parallelen Dateisystemen 2

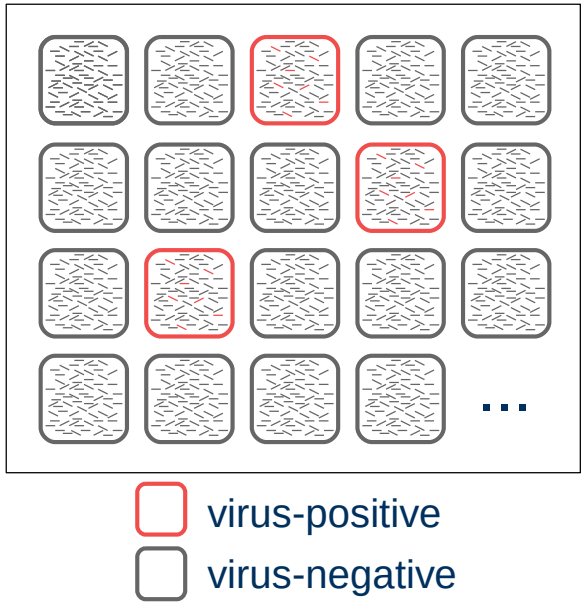
Genom-Datensätze

- Jeweils Millionen kurzer Sequenz-Fragmente
- Möglicherweise mit kleiner Teilmenge Virus-DNA unter der Host-DNA



Schritt 1, je 100 000 Datensätze

- Unterscheide Virus-positive und Virus-negative Datensätze



Schritt 2, 100 - 1000 Datensätze

- Rekonstruiere die vollständigen Virus-Genome aus den Virus-positiven Fällen

STEP	1) Download	2) Preprocess	3) Analyze	4) Postprocess
	• 100 to 100000 data sets per analysis	• move data depending on size	• wait for preprocess job	• clean data
TASKS / INFO	• 10-20 downloads in parallel	• preprocess data	• perform main analysis	• save results
	• submit preprocess job immediately after download	• hundreds of jobs may run in parallel	• hundreds of jobs may run in parallel	
FILE SYSTEM	/scratch	< 3 GB 3-25 GB > 25 GB	/tmp /beegfs/p_sra (/lustre/ssd) /scratch	data results
				/dev/null /projects/p_sra

Performance-Unterschiede durch separates Dateisystem

Vorteil durch separates Dateisystem:

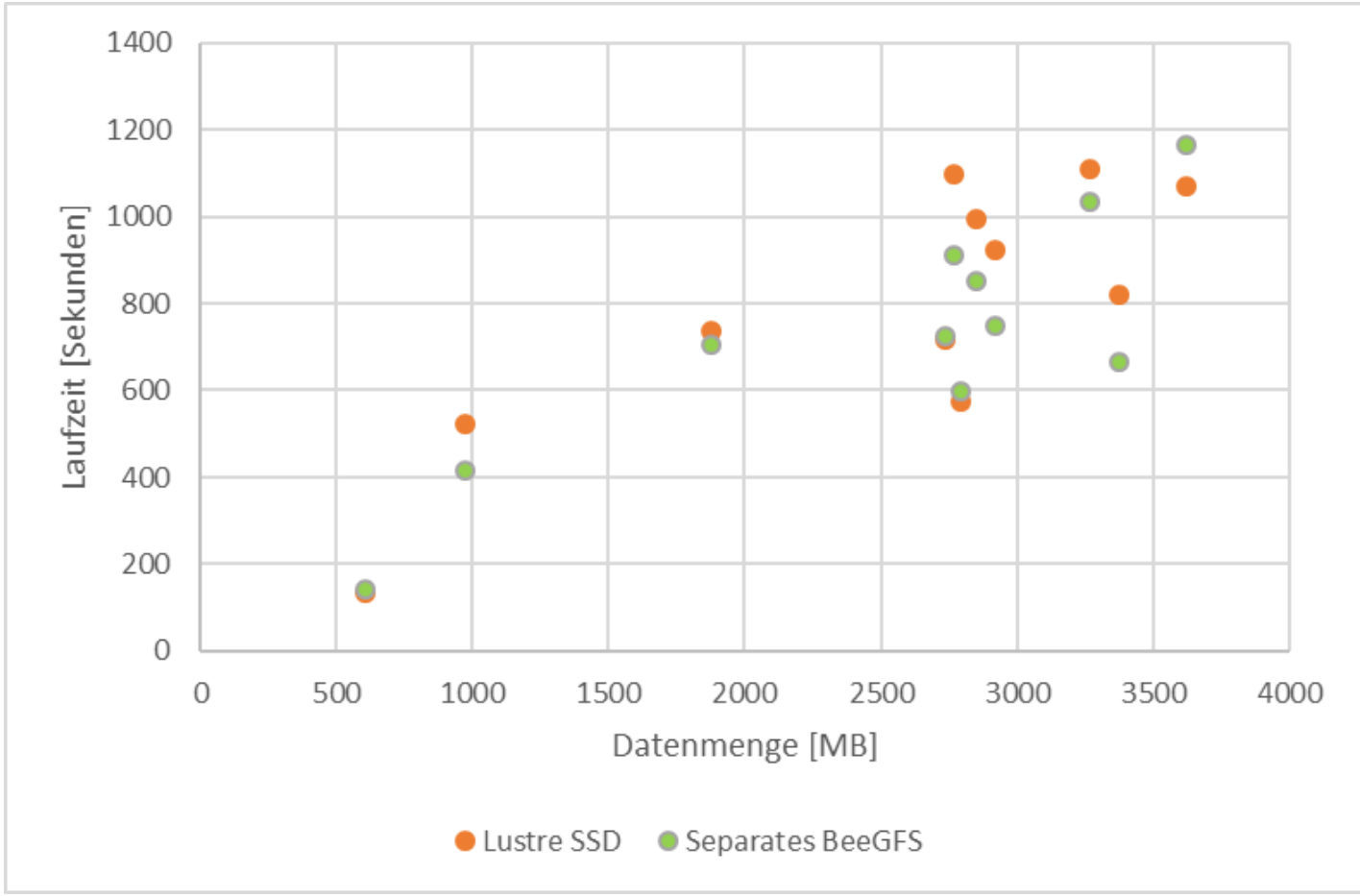
— 8,5 % geringere Laufzeit (in Summe)

— Schwankung zw. - 19,0 % und + 8.8 %

— Unterschiedliche Dateisysteme Lustre und BeeGFs

— Unterschiedliche Füllstände der Dateisysteme

— Verschieden viele Jobs der gleichen Arbeitsgruppe parallel (Scheduling der einzelnen Jobs)



Zusammenfassung

- Wenn umfangreiche I/O-Operationen in HPC-Anwendungen,
  dann typischerweise ausschlaggebend für die Gesamt-Performance
- Hardware entsprechend auslegen für hohe Storage- und Netzwerk-Bandbreiten
- Besonders I/O-intensive Anwendungen separieren
  - Eigenen Storage-Bandbreiten
  - Eigene Metadaten-Verwaltung
  - Trotzdem tlw. gemeinsam genutzte Transportwege im Netzwerk
  - Deutlich geringere Performance-Einflüsse von anderen / auf andere Anwendungen