30 KJ für 100 GB: Energieeffizientes Sortieren

Ulrich Meyer

Professur Algorithm Engineering

Woche der Informatik - Feb. 2010

Green Computing

Quelle: http://www.sxc.hu/photo/1255482

Quelle: www.sxc.hu/photo/107856

Ernster Hintergrund

- Wachsender Anteil der Energiekosten an Gesamtkosten
- Umweltbelastung durch Energieverbrauch / Abwärme

Weniger ernster Hintergrund

- Joulesort-Challenge: Spaß, Wettbewerb, Intellekt. Herausforderung

Energieverbrauch als neues Kostenmaß

http://www.sxc.hu/photo/1179339

Traditionell: reine Berechnungszeit

- Je schneller desto besser.
- Ziehe alle Register
  (schnelle CPU, großer Speicher, Multicores, . . .)

http://www.sxc.hu/photo/98930

Jetzt: Energieverbrauch

- Kürzere Berechnungszeit $\sim$ weniger Energie.
  $\Rightarrow$ optimiere Algorithmus und Implementierung.
- Schnellerer Rechner $\sim$ mehr Energie.
  $\Rightarrow$ finde beste Kompromiss-Hardware.

Joulesort Challenge

Teil eines etablierten Benchmarks: http://sortbenchmark.org/

Aufgabenstellung:

- Sort a fixed number of randomly permuted 100-byte records with 10-byte keys.
- The sort must start with input in a file on non-volatile store and finish with output in a file on non-volatile store.
- There are three scale categories for JouleSort: 10^8 (10GB), 10^9 (100GB), and 10^{10} (1TB) records
- The winner in each category is the system with the minimum total energy use.

Bsp: 30 KJoule = 30 000 Wattsek. entspricht z.B.

100 Watt für 5 Min. oder 20 Watt für 15 Min.

Joule 10^8 recs	2007, 8.6 kJoules **CoolSort** 11,600 records sorted / joule Mobile Core 2 Duo, 13 SATA laptop disks, Nsort Suzanne Rivoire (Stanford), Mehul A. Shah (HP Labs), Partha Ranganathan (HP Labs), Christos Kozyrakis (Stanford)	
Joule 10^9 recs	2007, 88 kJoules **CoolSort** 11,300 records sorted / joule Mobile Core 2 Duo, 13 SATA laptop disks, Nsort Suzanne Rivoire (Stanford), Mehul A. Shah (HP Labs), Partha Ranganathan (HP Labs), Christos Kozyrakis (Stanford)	2009, 87 kJoules **OzSort** 11,600 records sorted / joule 2.6 Ghz AMD Athlon LE-1640, 4GB RAM, 7x160 GB 7200 RPM SATA, Linux Nikolas Askitis and Ranjan Sinha Univ. Melbourne, Australia
Joule 10^{10} recs	2007, 2920 kJoules **CoolSort** 3,425 records sorted / joule Mobile Core 2 Duo, 13 SATA laptop disks, Nsort Suzanne Rivoire (Stanford), Mehul A. Shah (HP Labs), Partha Ranganathan (HP Labs), Christos Kozyrakis (Stanford)	

Resultate von 2009.

Warum gerade Sortieren ???

Sortieren von Anfang an . . .

Quelle: www.sxc.hu/photo/250384

Mama Räum endlich dein Zimmer auf!

Kind Wieso ???

Mama Damit Du besser suchen kannst.

Kind Ich mach doch Hashing . . .

Mama Was ??? Papa, sag' Du doch auch mal was!

Papa Geht dieses Hashing auch in meiner Werkstatt?

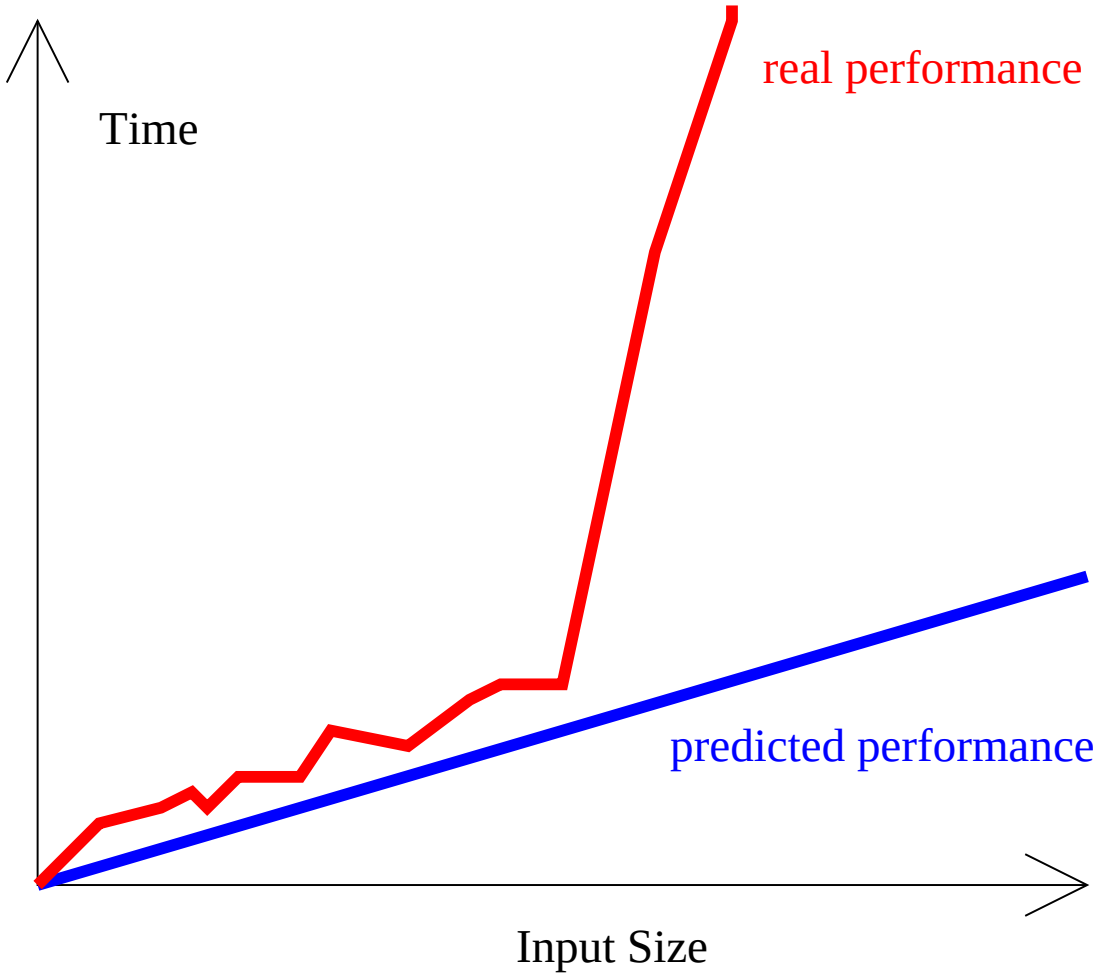
Mama Typisch Mann, AUFRÄUMEN !!!

Kind [räumt widerwillig auf]

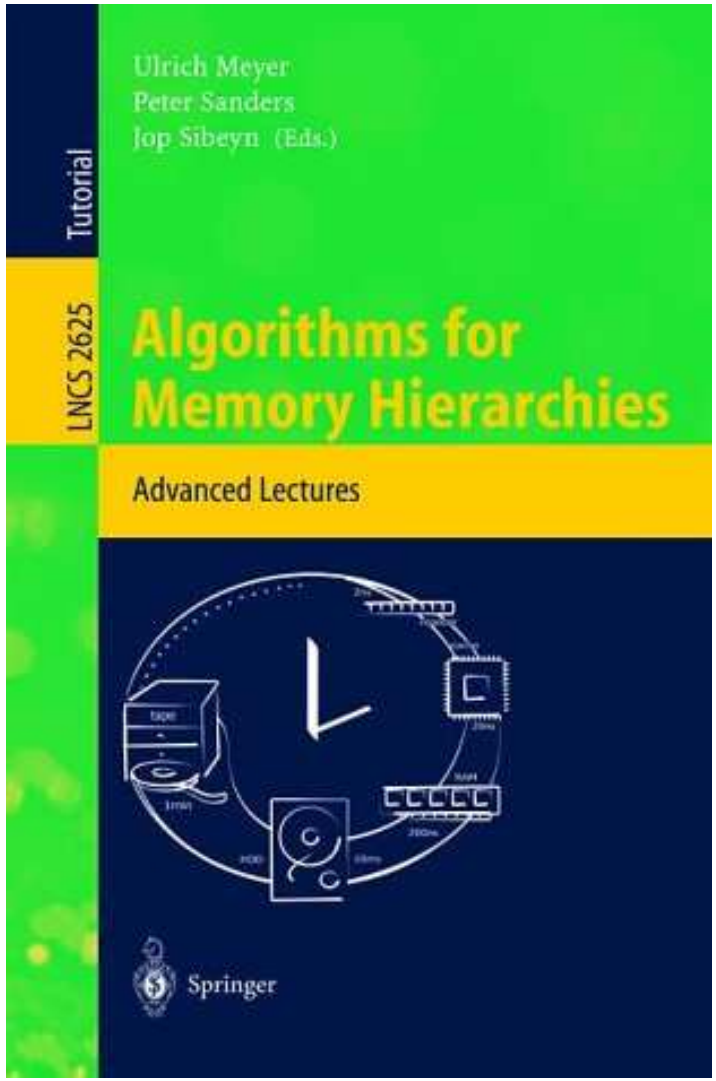
Circa 25% aller Rechenzeit im kommerziellen Bereich entfällt auf das Sortieren von Daten.

Eine Aufwärmübung (noch ohne Sortieren)

```
int[1..n] X,Y,Z; // Z Permutation von 1..n
for i=1 to n do X[i]:=Y[Z[i]];
```

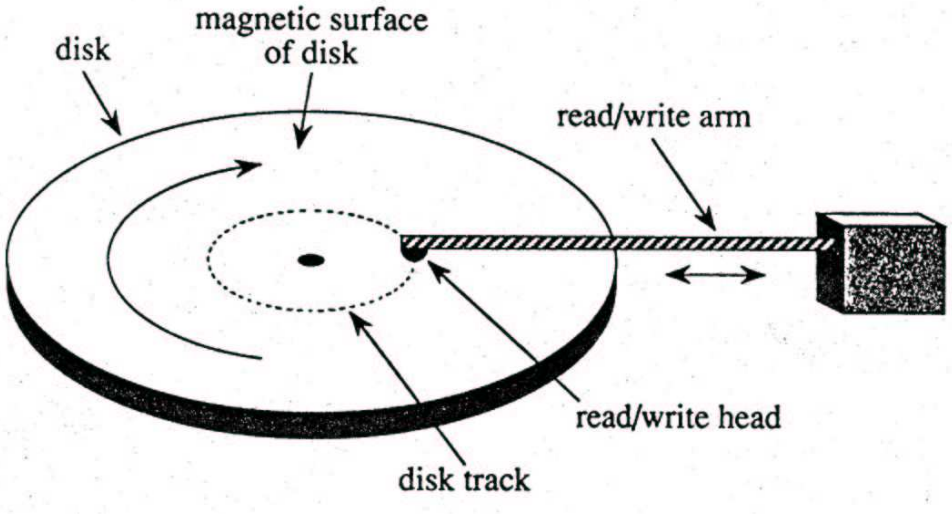


Speicherhierarchien



- Traditionell:
  Uniforme Kosten für Speicherzugriffe.
- Heute/Zukunft:
  Kosten hängen davon ab, wo die Daten gespeichert sind.
- 5-7 Hierarchiestufen: Register, L1-L3 Cache, Hauptspeicher, Festplatten, Magnetbänder
- Verzögerungen: Faktor $10^1 - 10^7$.
- Explizite Optimierung der Speicherzugriffe für höchste Performance oft unabdingbar.

Warum sind Festplatten so langsam?



Quelle: J.S. Vitter

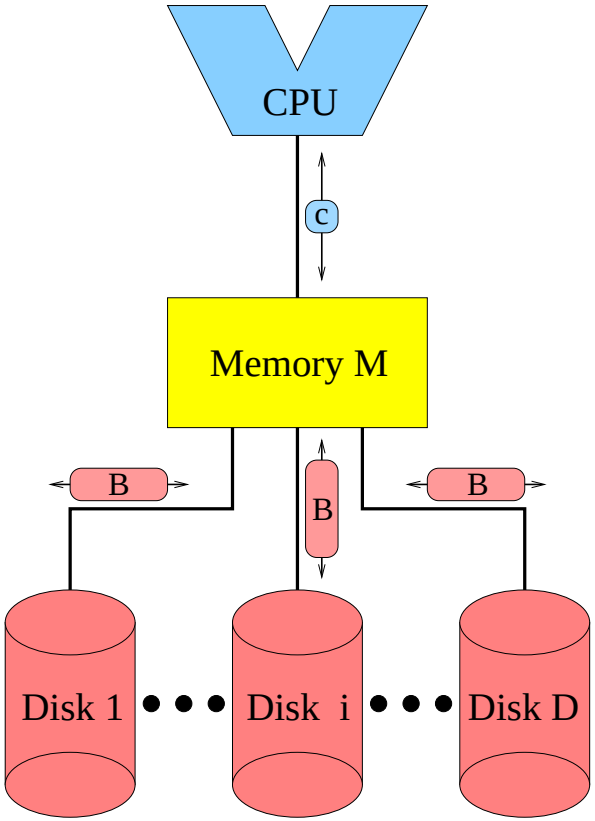
Quelle: www.sxc.hu/photo/1144732

Bestandteile der Zugriffskosten:

- Suchzeit (Millisekunden, LANGSAM)
- Rotationsverzögerung (Millisekunden, LANGSAM)
- Lesezugriff (Nanosekunden, SCHNELL)

Das Lesen vieler aufeinanderfolgender Daten dauert kaum länger als das Lesen eines einzigen Datums. $\Rightarrow$ Benutze Blocktransfers.

Externspeicher (EM) Modell [VS94]



- Hauptspeicher $M \ll$ Problem N
- Sekundärspeicher = D Festplatten
- Datentransfer in Blöcken der Größe B ($\sim 10^5$)
- $\leq D \cdot B$ Daten pro I/O-Schritt ($\sim 10^2$ pro Sek.)
- Ziel: minimiere I/O
- $\mathrm{scan}(x) := \mathcal{O}(\frac{x}{D \cdot B})$ I/Os.
- $\mathrm{sort}(x) := \mathcal{O}(\frac{x}{D \cdot B} \cdot \log_{M/B} \frac{x}{B})$ I/Os.

Unsere Aufwärmübung (jetzt mit Sortieren)

```
int[1..n] X,Y,Z; // Z Permutation von 1..n
for i=1 to n do X[i]:=Y[Z[i]];
```

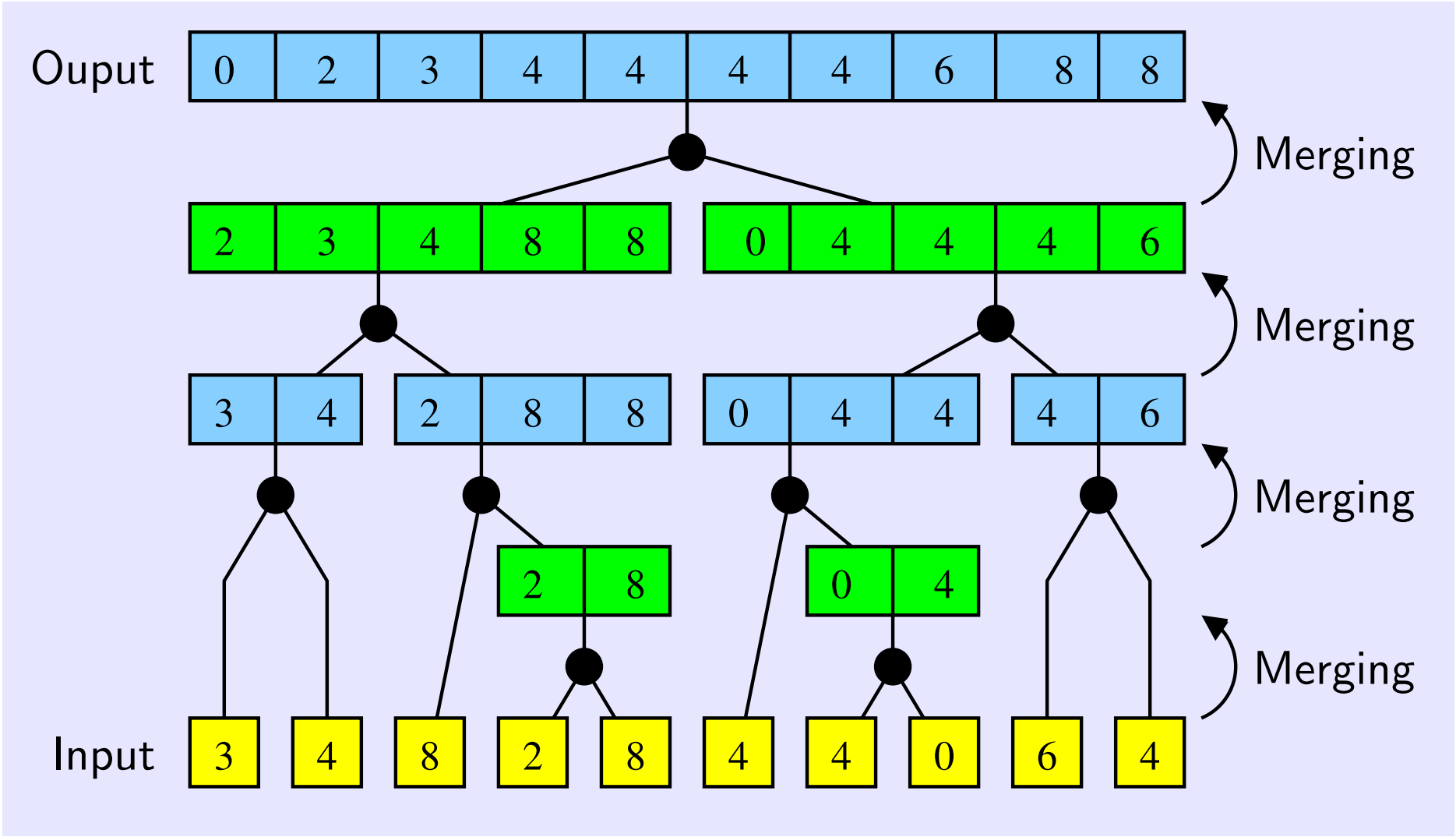
- Schlimmster Fall: $\Omega(n)$ I/Os.

Besser:

```
SCAN Z:     (Z[1]=17,1),     (Z[2]=5,2),      ...
SORT(1st): (Z[73]=1,73),    (Z[12]=2,12),    ...
par SCAN : (Y[1],73),       (Y[2],12),       ...
SORT(2nd): (Y[Z[1]],1),     (Y[Z[2]],2),     ...
par SCAN :  X[1]:=Y[Z[1]],  X[2]:=Y[Z[2]], ...
```

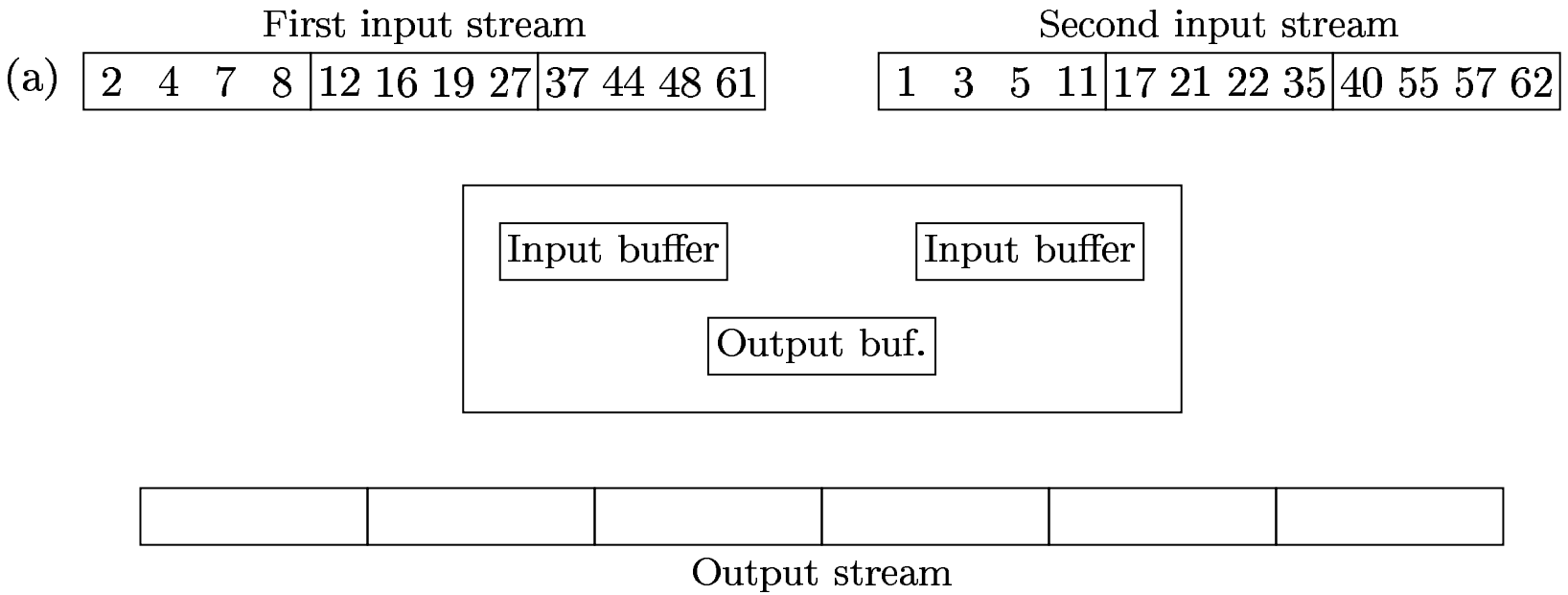
- Schlimmster Fall: $\mathcal{O}(\mathrm{sort}(n))$ I/Os.

Sortieren mit Mergesort



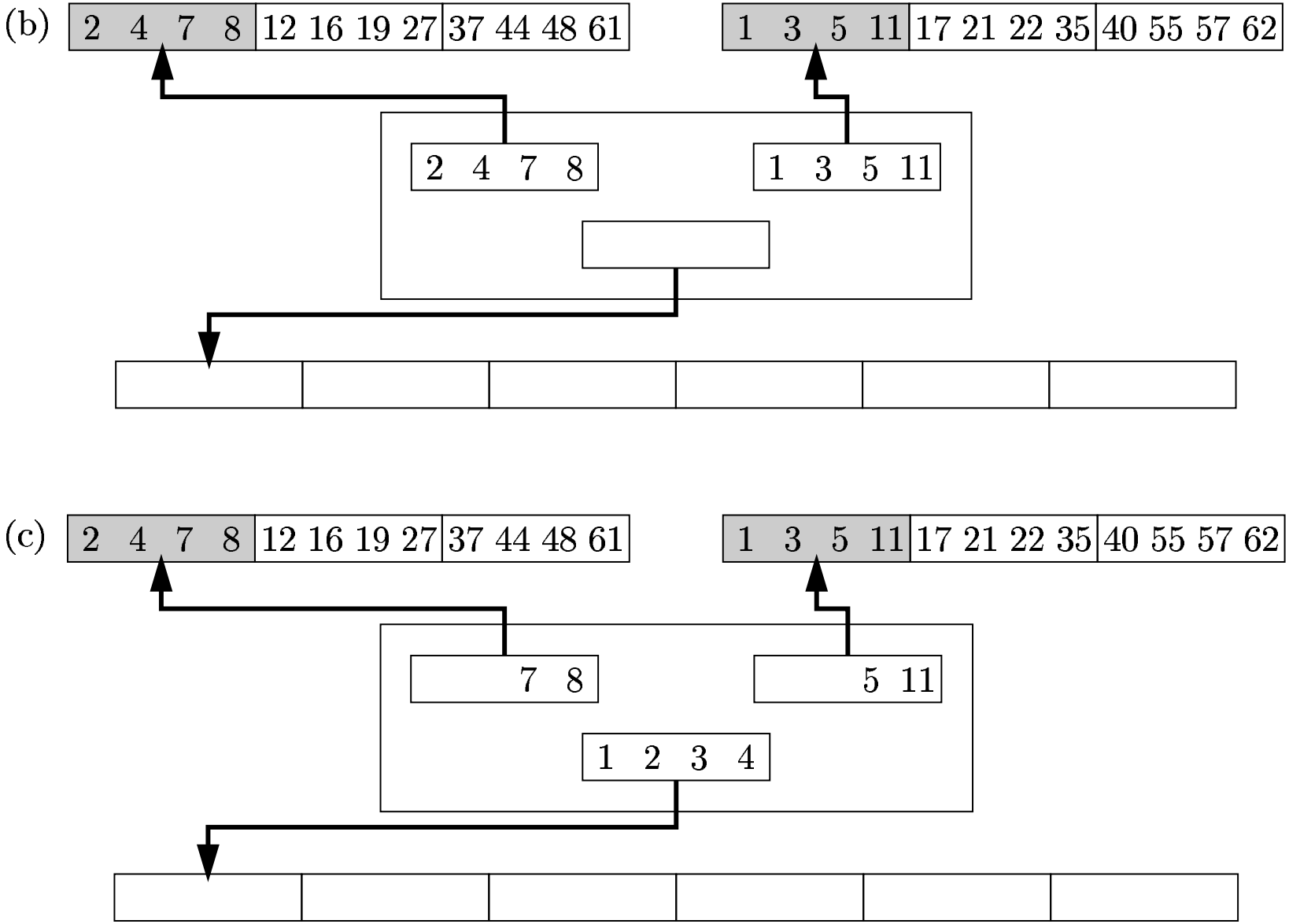
Quelle: Gerth Brodal

I/O-effizientes Mergesort



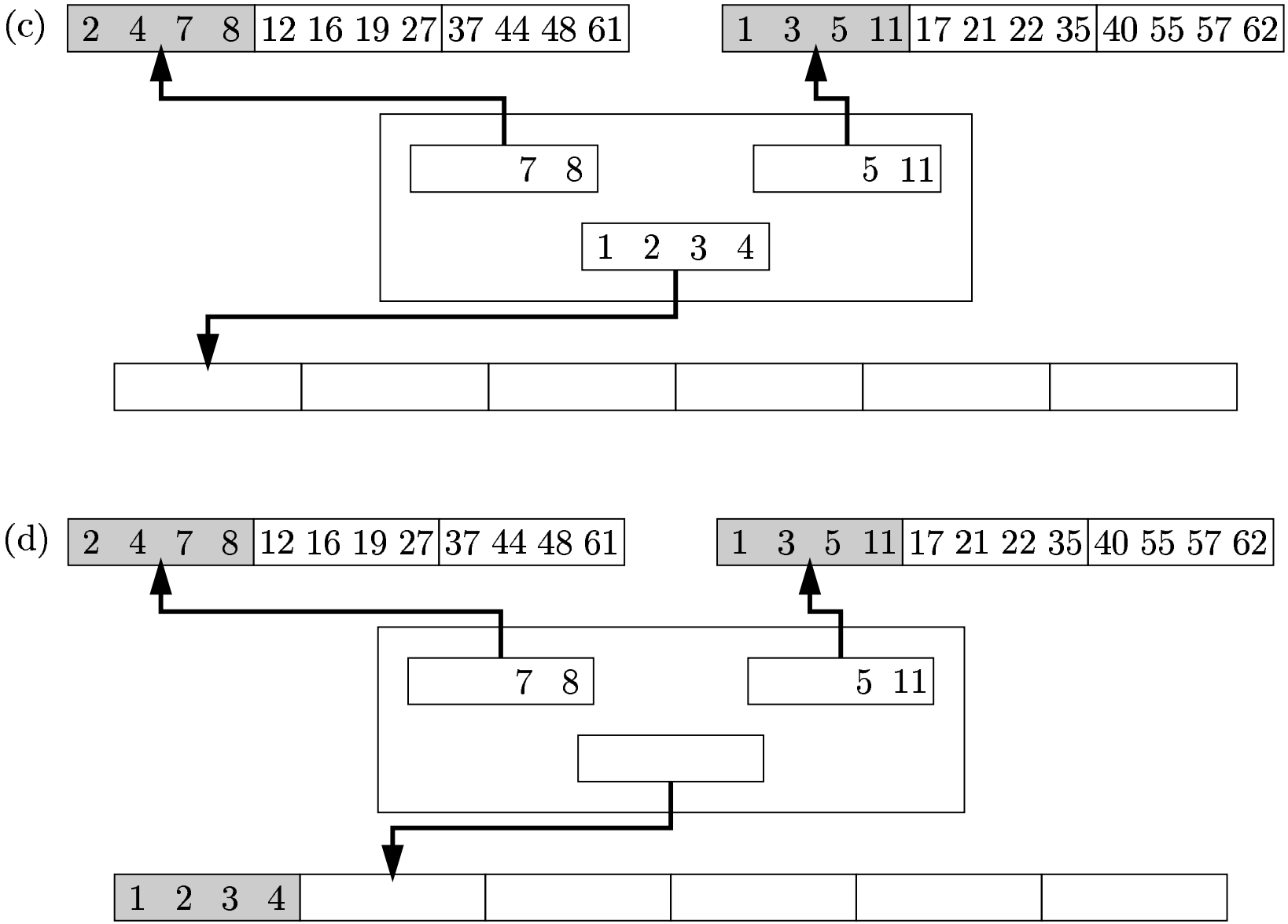
Quelle: A. Maheshwari / N. Zeh

I/O-effizientes Mergesort



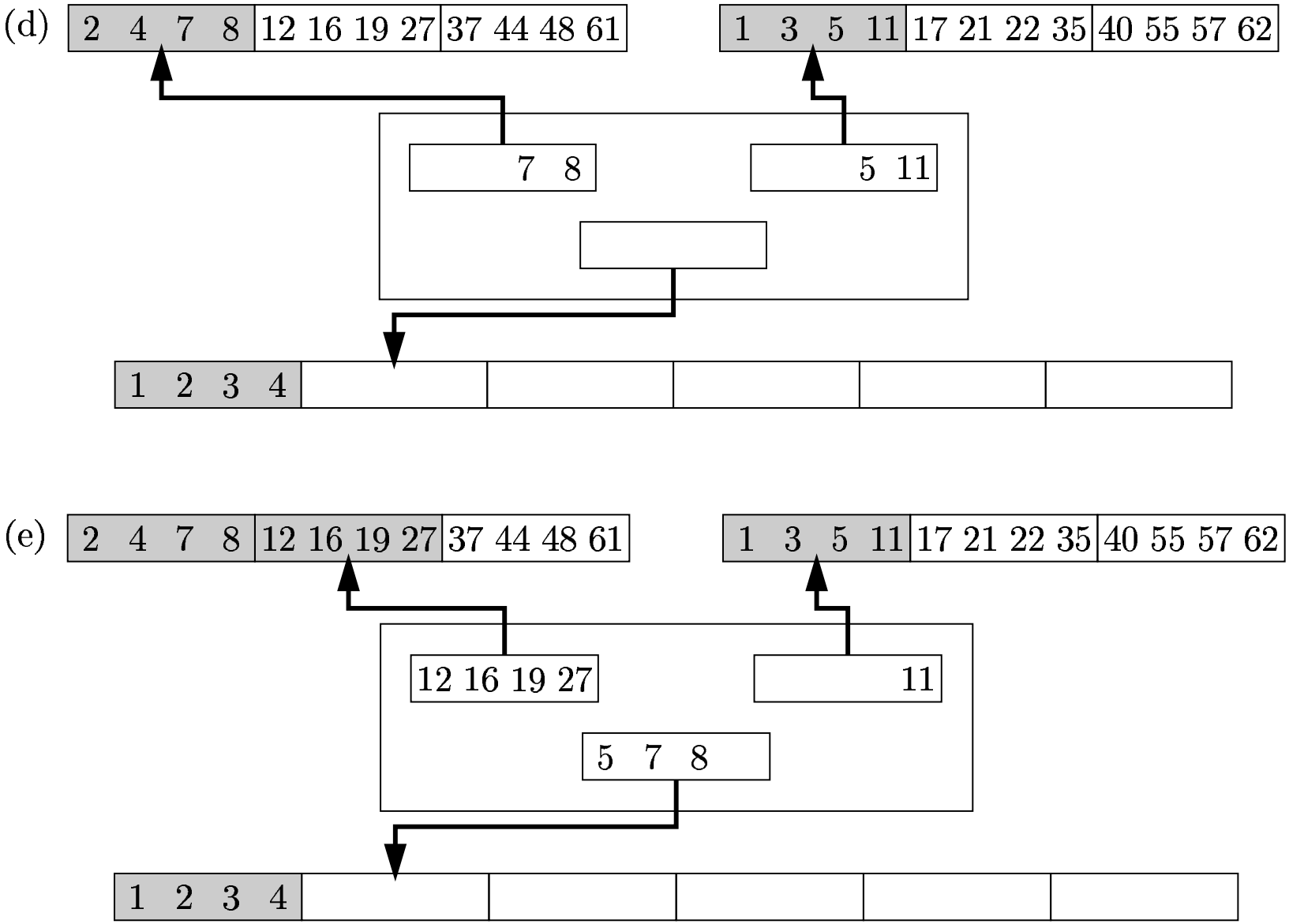
Quelle: A. Maheshwari / N. Zeh

I/O-effizientes Mergesort



Quelle: A. Maheshwari / N. Zeh

I/O-effizientes Mergesort



Quelle: A. Maheshwari / N. Zeh

I/O-Performanz von Mergesort

Beispiel: $N = 10^9$, $M = 10^7$, $B = 10^4$ Elemente:

Standard-Mergesort: $\mathcal{O}(N/B \cdot \log_2(N))$ I/Os.
30 Merge-Phasen.

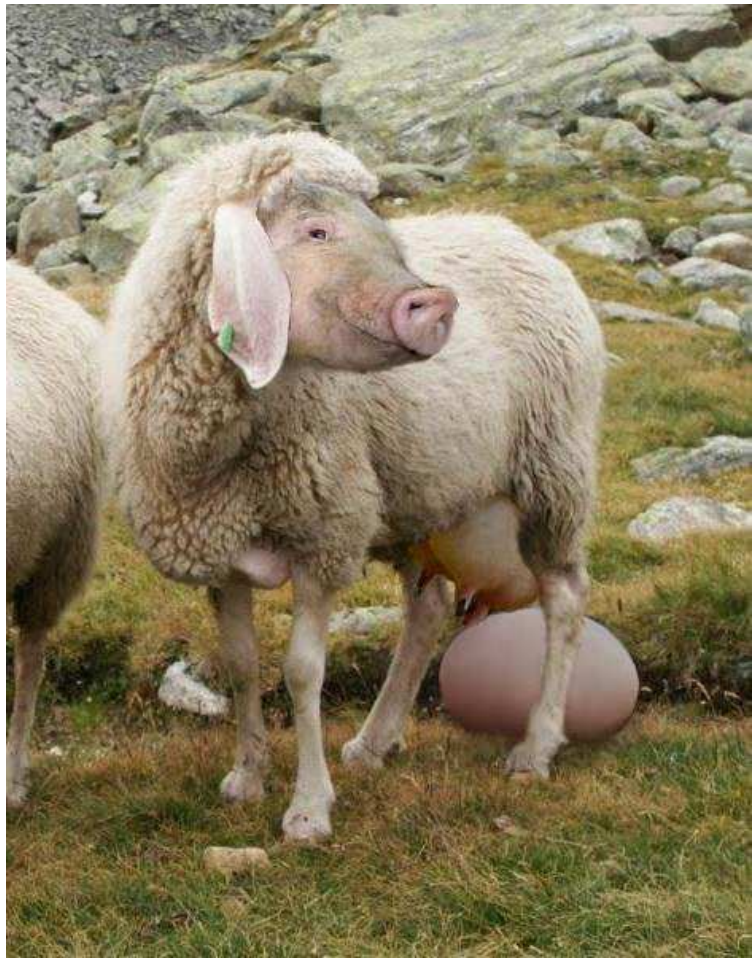
Mit Vorsortieren von Teilmengen der Größe $\Theta(M)$: $\mathcal{O}(N/B \cdot \log_2(N/M))$ I/Os.
7 Merge-Phasen.

Zusätzlich mit $\Theta(M/B)$-fach Mischen: $\mathcal{O}(N/B \cdot \log_{M/B}(N/M))$ I/Os.
1 Merge-Phase.

Viele Alternativen/Erweiterungen:

- Distribution Sort $\approx$ I/O-eff. Quicksort
- Sortieren mit I/O-eff. Prioritätswarteschlange
- Parallele Disks (nutzen wir)
- Überlappen von I/O und Berechnung (nutzen wir)
- Sortieren ist zentraler Bestandteil der meisten I/O Bibliotheken
- ...

Bau eines Energie-effizienten Sortierrechners . . .



Quelle: www.zachseinblog.de/wp-content/uploads/2009/10/Wollmilchsau.jpg

Schnell und trotzdem sparsam bzgl.

- CPU/Cores
- Chipset
- I/O-Controller
- Externspeicher
- Kühlung
- . . .

Die eierlegende Wollmilchsau:

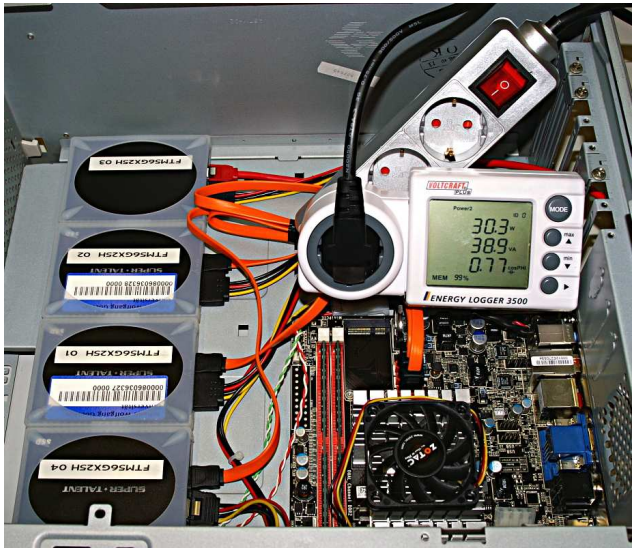
ein balancierter Kompromiss !!

Side by Side . . .

Quelle: csl.stanford.edu/∼christos/pics/coolsort.png

Joulesort Gewinner 2007–2009
[Rivoire etal. (Stanford & HP Labs)]:

- Intel Core 2 Duo Mobil-CPU
- viele Notebook-Festplatten



Unser Ansatz [Beckmann etal.
(GU Frankfurt & KIT Karlsruhe)]:

- Atom N330 CPU
- wenige Flash-Speicher (SSDs)

Side by Side – Details . . .

	Bisheriger Rekord [Rivoire etal. 07]		Unser Rechner	
Component	**Type**	**TDP**	**Type**	**TDP**
Processor	CPU Intel Core 2 Duo T7600 Mobil-CPU	34 W	Intel Atom N330 2 cores, 4 threads	8 W
Memory	Kingston 2x1 GiB	4 W	Kingston 2x2 GiB	4 W
Board	Asus N4L-VM DH	n/a	Zotac IONITX-A	12 W
I/O	2x PCIe to SATA HighPoint Rocket RAID	12 W	4x SATA 3.0 Gibps onboard	–
Disks	13x Hitachi TravelStar 5K160 Notebook HDs	23 W	4x Super Talent FTM56GX25H SSDs	4 W
Fan		n/a		1 W
OS drive		–	USB-Stick	1 W
Estimated Total (net)				30 W
Estimated Total (overall)				37.5 W
Typical Idle (overall)		60 W		25 W
Typical Loaded (overall)		100 W		37 W

Die große Frage . . .

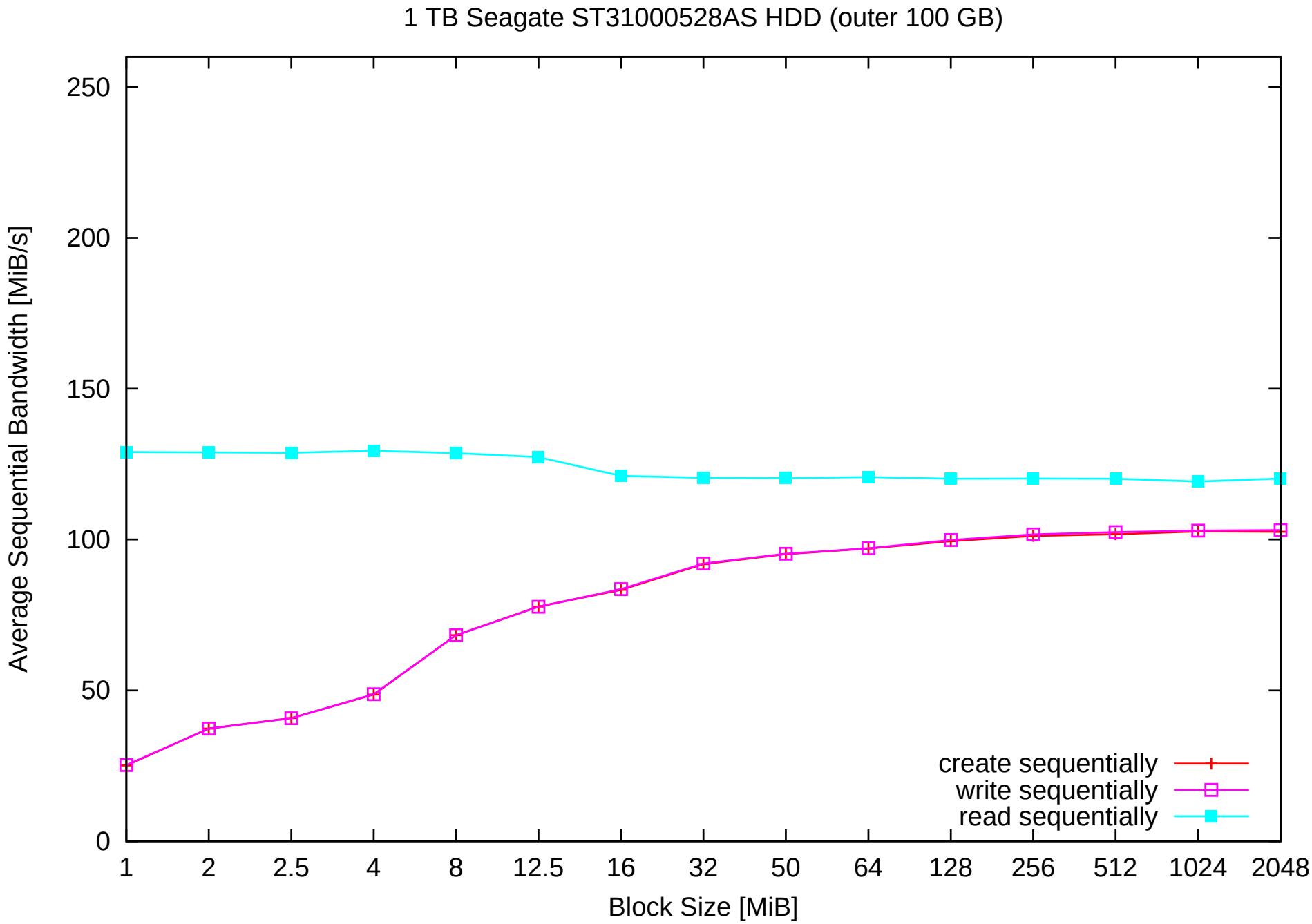
Unsere Maschine ist zwar viel sparsamer, aber ist sie auch viel langsamer ???

Quelle: www.sxc.hu/photo/1155466

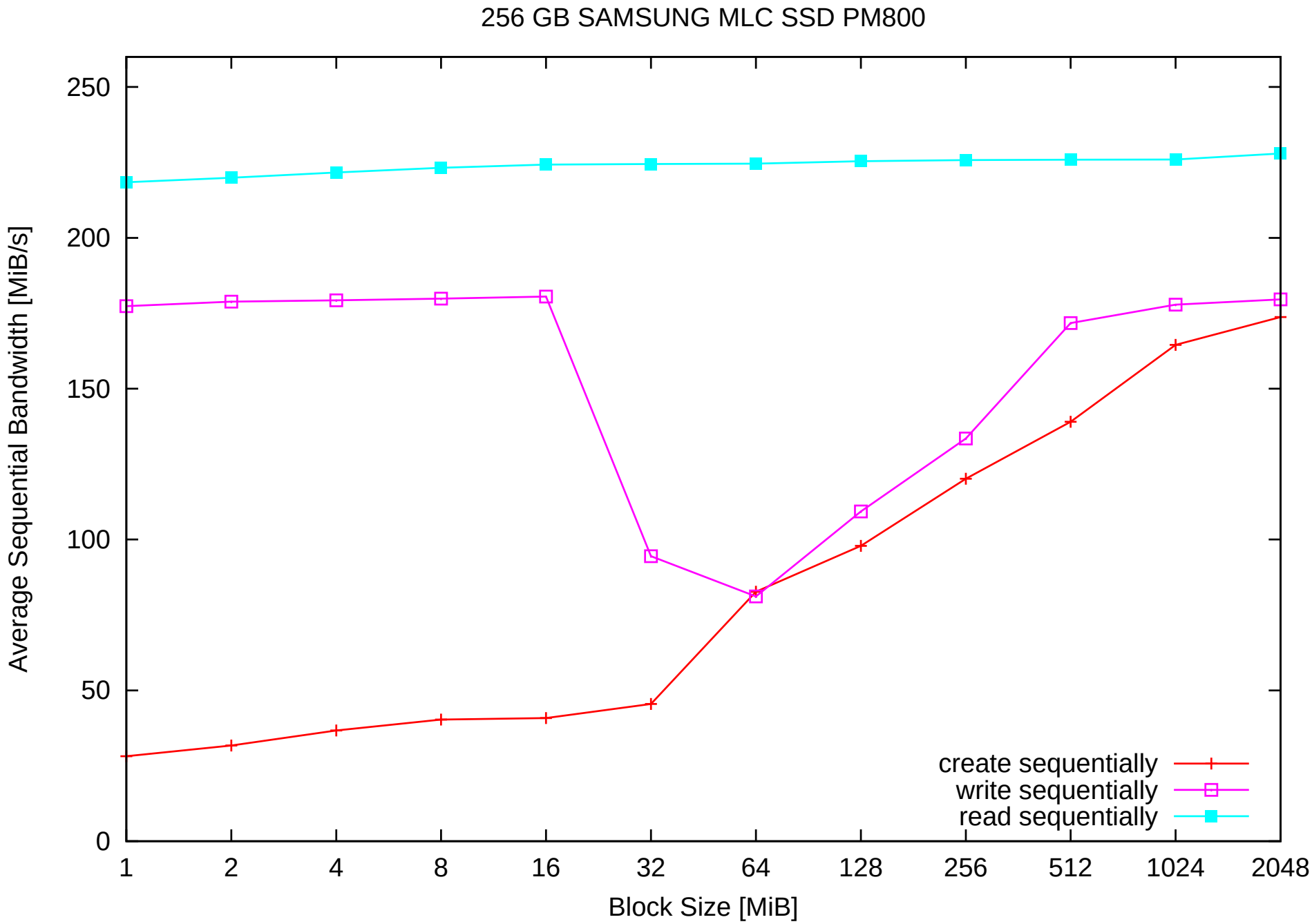
Zentral: Performanz von
Solid State Disks gegenüber Hard Disks

- SSDs ohne Mechanik
- stürmische Entwicklung
- komplizierte Controller
- schwerer modellierbar/vorhersagbar
- eigenes DFG-Projekt

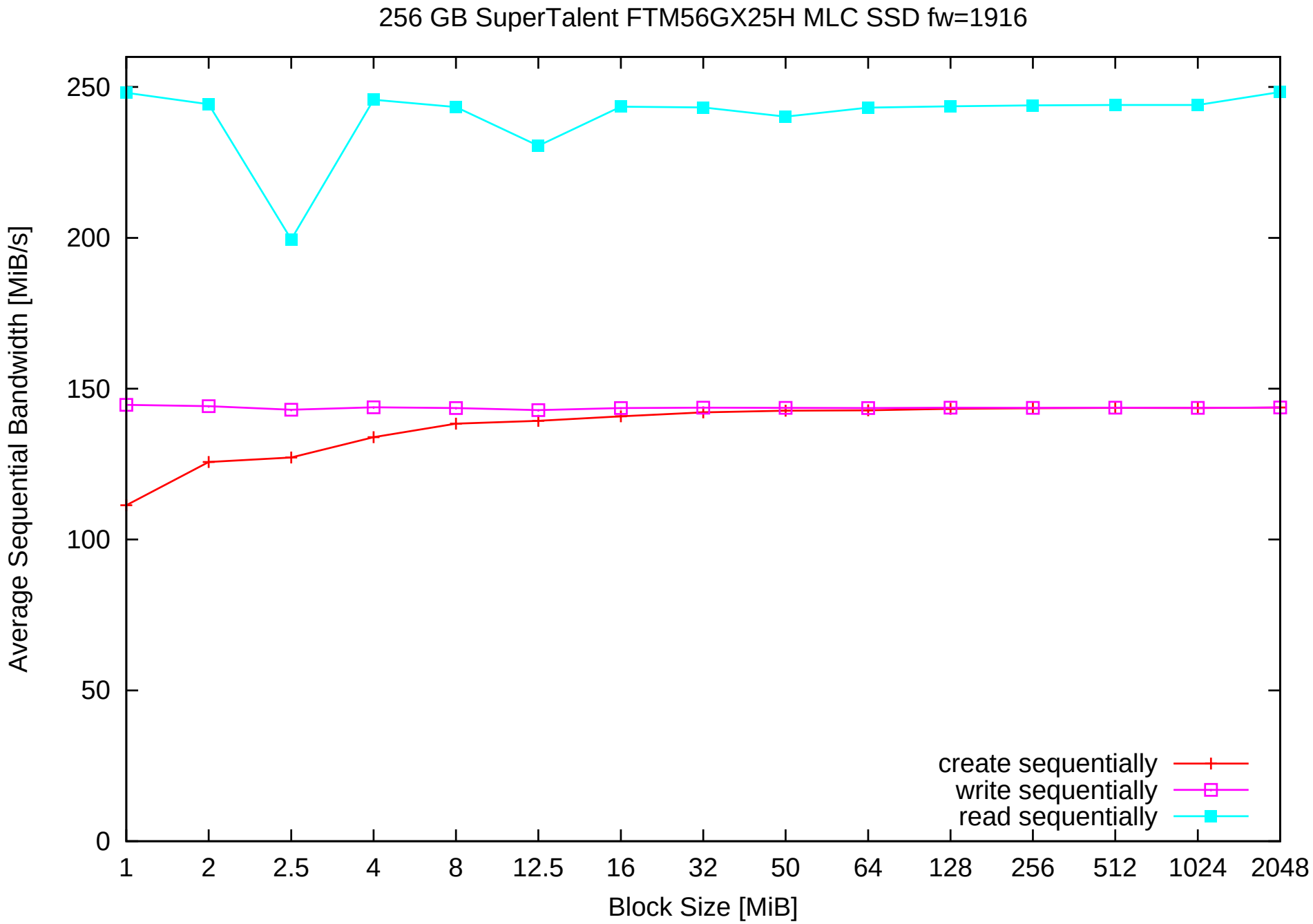
Leistung einer Server-Festplatte



Leistung einer ’problematischen’ SSD



Leistung einer guten SSD



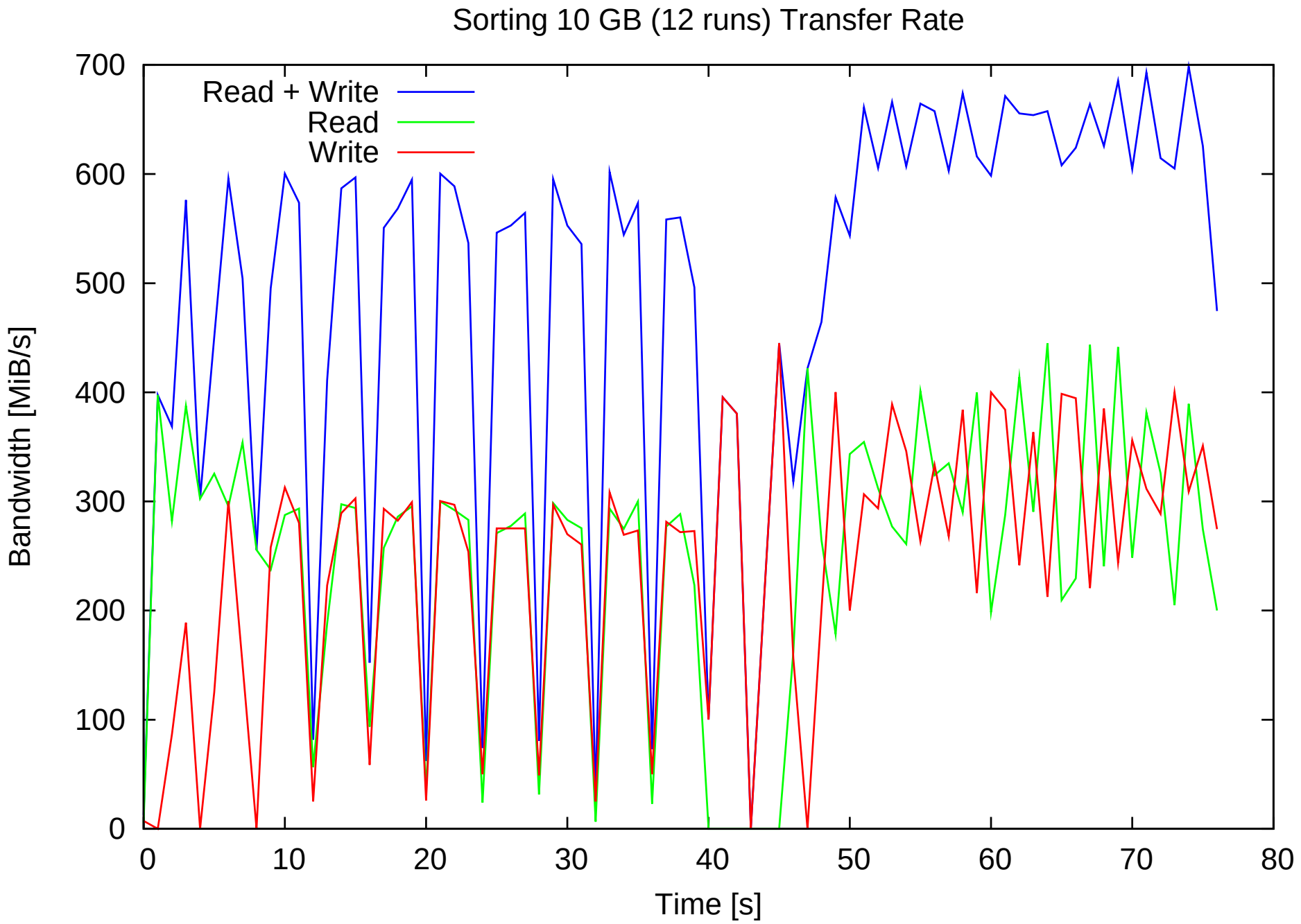
Zwischenfazit SSDs

Mit ’richtigem’ Controller und bei einfachen Zugriffsmustern sind SSDs klassischen Festplatten deutlich überlegen.

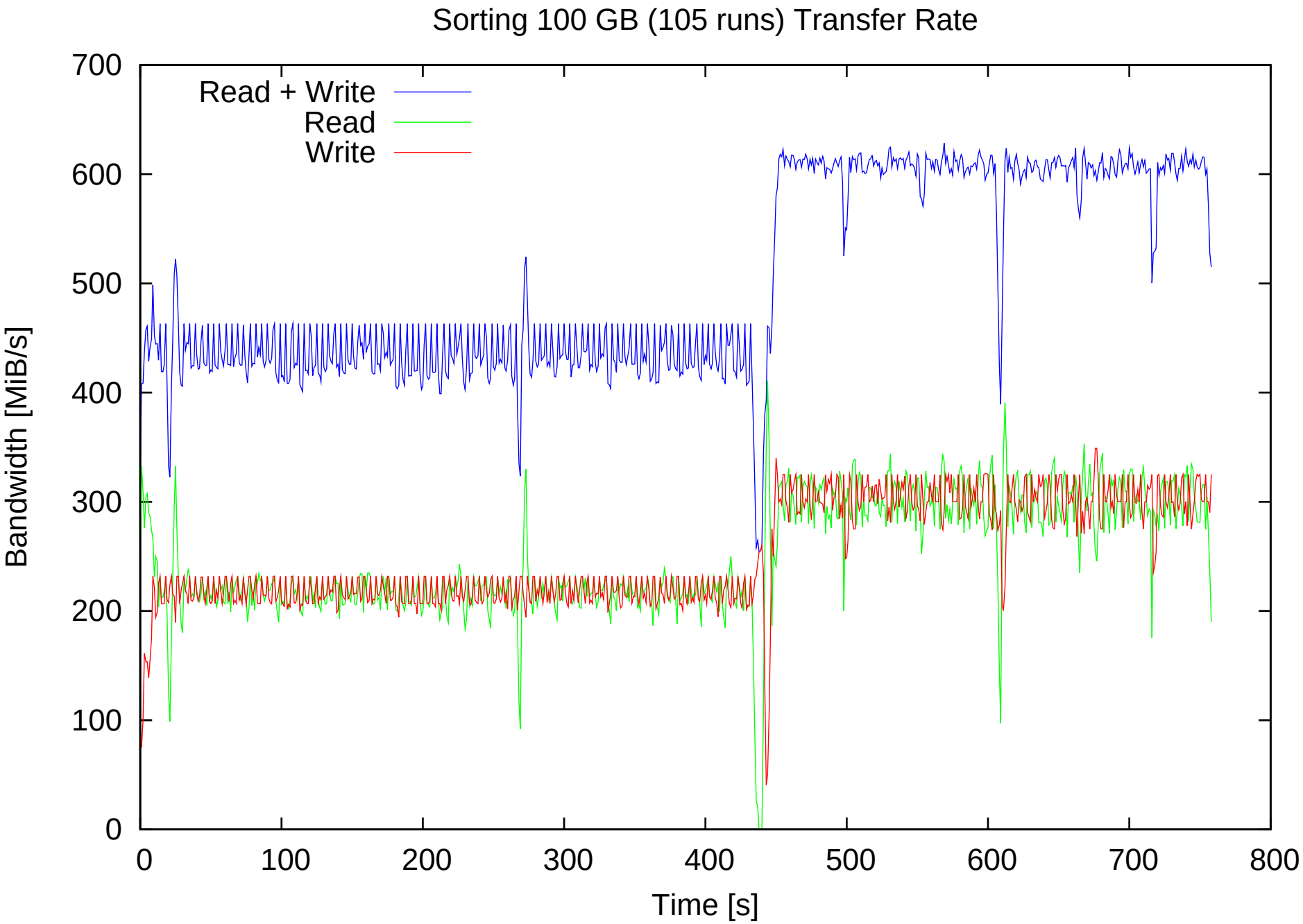
Offene Fragen:

- Was passiert bei gemischten Schreib-/Lese-Zugriffen in echten Algorithmen?
- Wie verhalten sie sich in einem RAID?

SSD-Durchsatz beim 10 GB Problem [4-er RAID, ∅: 1 sec]



SSD-Durchsatz beim 100 GB Problem [4-er RAID, ∅: 4 sec]



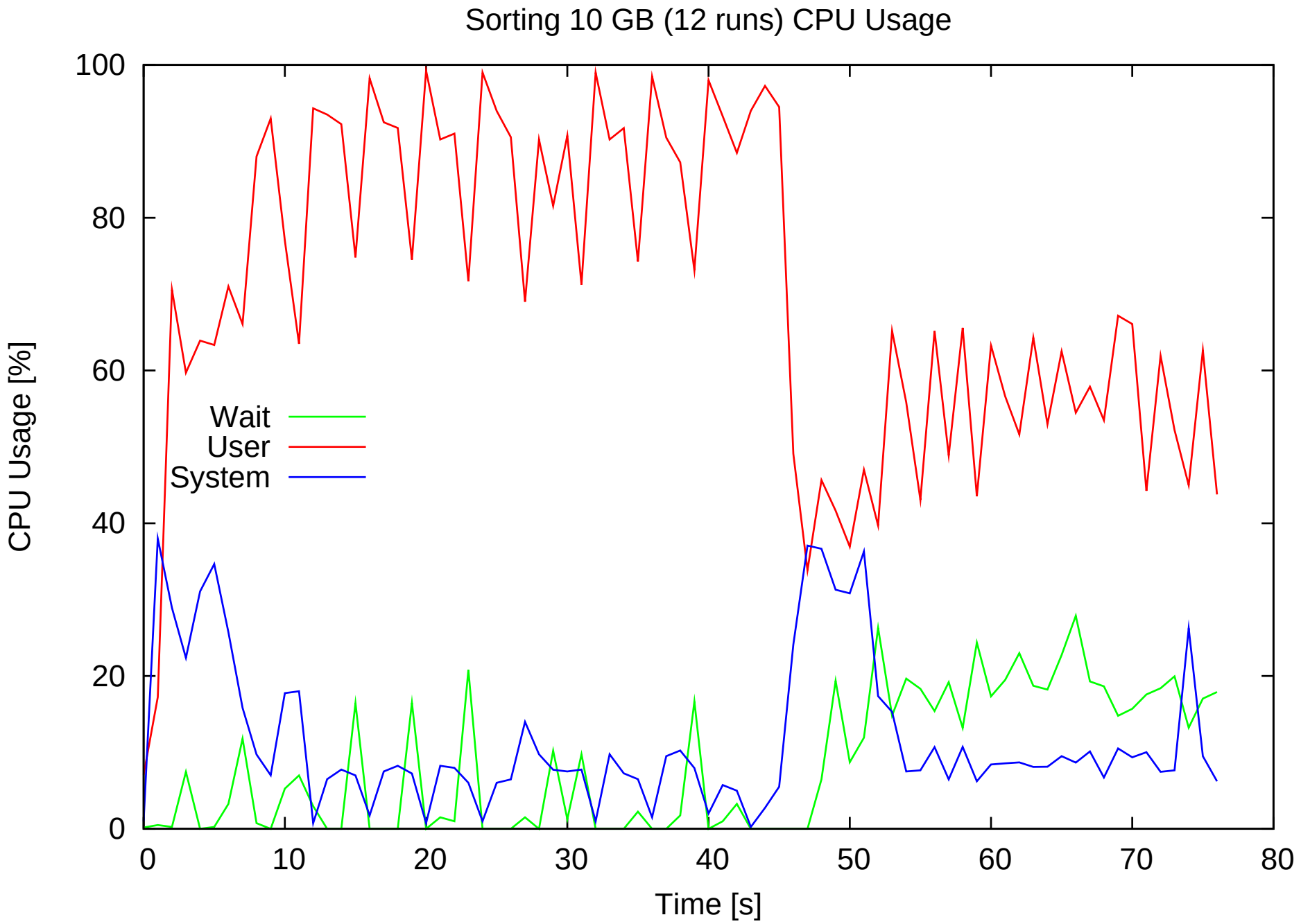
I/O und Berechnung

Im besten Fall ausgeglichenes Verhältnis/
Überlappung zwischen I/O und CPU-Last.

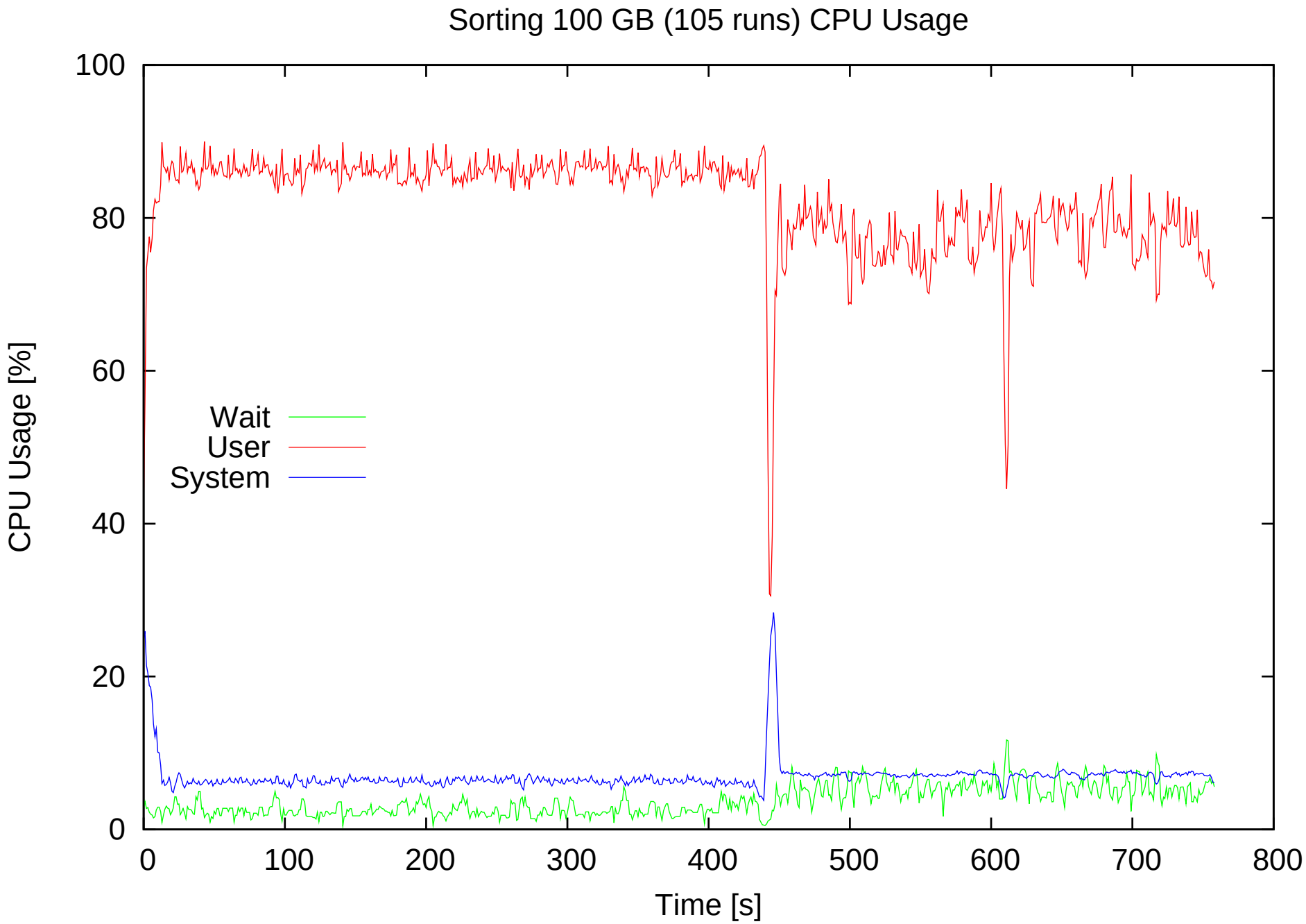
Kennzeichen:

- dauerhaft hoher I/O-Durchsatz
- geringe und seltene I/O-Wartezeit
- dauerhaft hohe User-Load

CPU-Load beim 10 GB Problem [∅: 1 sec]



CPU-Load beim 100 GB Problem [Ø: 4 sec]



Joulesort Ergebnisse und Ausblick

Kategorie	Zeit	∅-Leistung	Energie	Elemente/J	Verbesserung
10 GB	77 s	36.8 W	2.8 kJ	35 453	≥ 3
100 GB	756 s	36.4 W	27.5 kJ	36 381	≥ 3
1000 GB	21 906 s	33.0 W	724.0 kJ	13 818	≥ 4

Offene Fragen:

- Energie-effiziente Algorithmen für andere Probleme.
- Kostenmaß 'Anschaffung plus Energieverbrauch'.

Weitere Infos: www.ae.cs.uni-frankfurt.de

Unser Joulesort-Rechner