# SSDs als Cache für HDDs

## CacheCade vs. BCache

Dirk Geschke

Linux User Group Erding

23. Oktober 2013

# Gliederung

# Allgemeines

- Festplatten existieren schon länger, haben aber Vor- und Nachteile
- SSDs sind neu, hier sind Vor- und Nachteile getauscht
- Idee: Kombination von beiden Systemen
- 2 Vertreter: CacheCade (LSI) und BCache (im Linux-Kernel enthalten)

# Vor- und Nachteile

## Vorteile:

+ große Speicherkapazität, derzeit 4 TB pro HDD möglich
+ sehr günstiger Preis pro GB
+ hohe Lebensdauer
+ bewährte Technik, Schwächen sind bekannt

## Nachteile:

- rotierende Scheiben
- bewegliche Teile
- lange Zugriffszeiten
- relativ langsam

# Ausweg aus HDD-Problemen

- Einsatz von mehren Platten als eine virtuelle HDD: RAID
- Verschiedene Varianten sind möglich, die wichtigsten sind:

  RAID 0 Striping, sehr schnell $\Longrightarrow$ kein Ausfall eine HDD tolerabel

  RAID 1 Mirroring, langsam $\Longrightarrow$ Ausfall einer HDD-Hälfte tolerabel

  RAID 5 verteilte Paritätsinformationen $\Longrightarrow$ Ausfall einer HDD tolerabel

  RAID 6 wie RAID 5 plus weitere Paritäts-HDD $\Longrightarrow$ Ausfall zweier HDDs tolerabel

- Kombinationen möglich: RAID 10, RAID 50, RAID 60, ...
- Hotspare-Platten sinnvoll, sofern RAID es erlaubt
- Aber: Stripe-Size, Parity-Berechnungen, Rebuild-Zeiten

# Vor- und Nachteile

## Vorteile:

- \+ extrem kurze Zugriffszeiten
- \+ sehr schneller Durchsatz
- \+ keine Mechanik, leise, vibrationsarm, stoßfest, kein Verschleiß
- \+ geringer Stromverbrauch (Notebook!)
- \+ kaum Wärmeentwicklung, temperaturunempfindlich

# Vor- und Nachteile

## Nachteile:

- Preis pro GB sehr hoch
- Größe im Vergleich zu HDD recht klein
- Lebensdauer, vor allem Art der NAND-Flash Bausteine relevant, betrifft nur das Schreiben von Blöcken
- Firmware oft heikel (*wear leveling*)
- TRIM-Support hilfreich
- Umdenken notwendig
  $\Longrightarrow$ *read-ahead*, scheduler, Dateisysteme, etc.

# Kombination der Systeme

- Vorteile der SSDs mit den Vorteilen der HDDs kombinieren
- Nachteil: SSDs sind nur noch Cache!
- *WriteBack* für Schreiben notwendig:

  Problem Lebensdauer
  Lösung RAID $>$ 0
- problematisch im Notebook -> oft nur *WriteThrough*
- Vorteil beim Notebook: Schnelles booten!

## Preisfrage: Lohnt es sich?

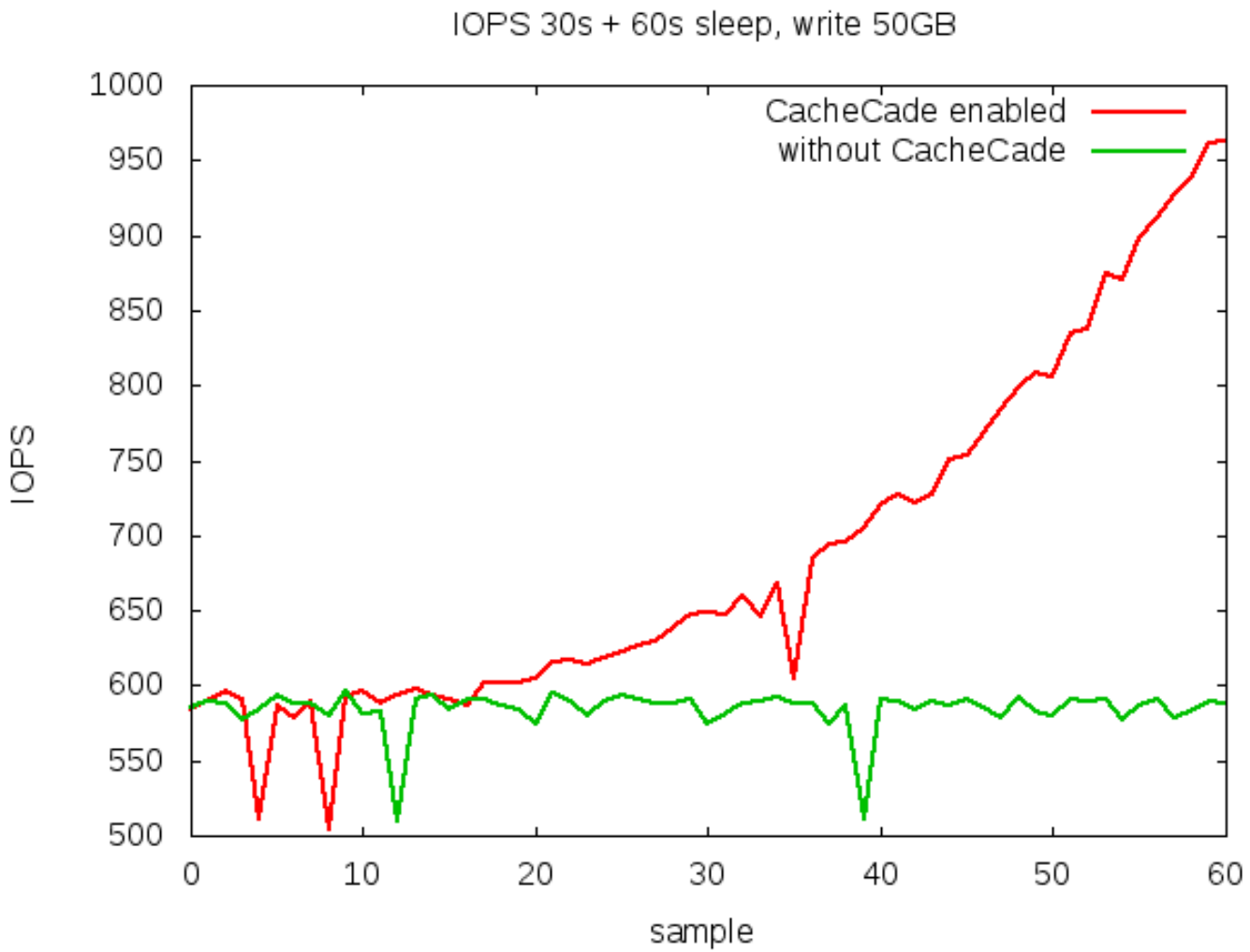
# Testsetup

- RAID-Controller von LSI: 9266-8i mit CacheCade
- 10 HDDs à 3 TB (2794 GiB) Hitachi HUA 72303
- 8 SSDs à 250 GB (238 GB) OCZ-VERTEX 4
- 2 SSDs à 120 GB INTEL für CacheCace-Tests (zertifiziert)
- Host: 2 CPUs Xeon CPU E5-2690 à 8 Cores + HT, 256 GB RAM
- Testprogramm: **sdb** – **S**imple **D**isk **B**enchmark und O_DIRECT
- Verwendete Blockgröße für IOPS: 512 Bytes oder 4096 Bytes

# CacheCade

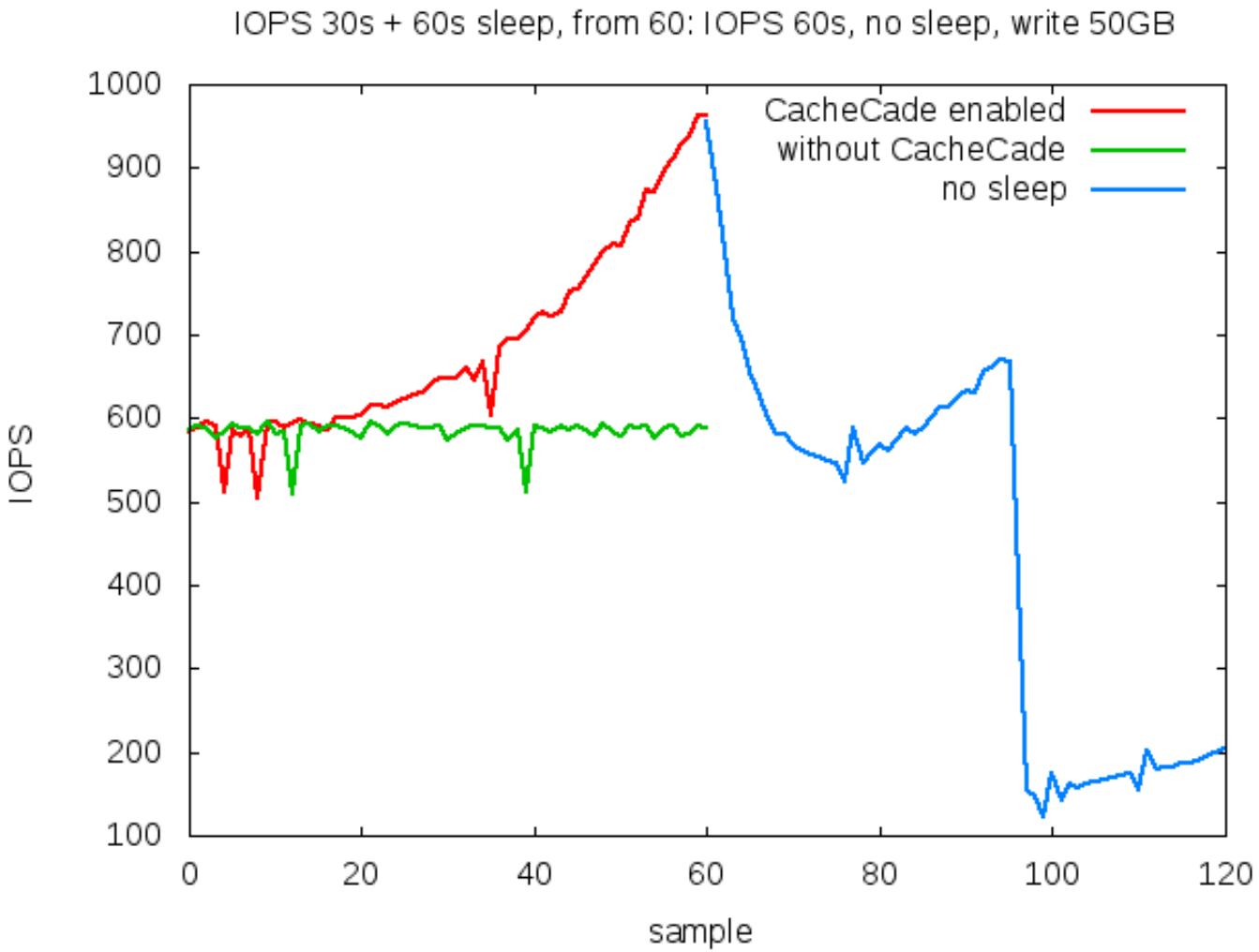
- kommerzielles Produkt von LSI in Verbindung mit RAID-Controller
- wird vom RAID-Controller gesteuert
- derzeit begrenzt auf 512 GB
- nur RAID 0/1/10(1e) möglich
- benötigt Lernphase

# Vergleich mit und ohne CacheCade, OCZ-Disks




- Gut zu beobachten: Lernphase!

# Vergleich CacheCade, ohne Pause, OCZ-Disks

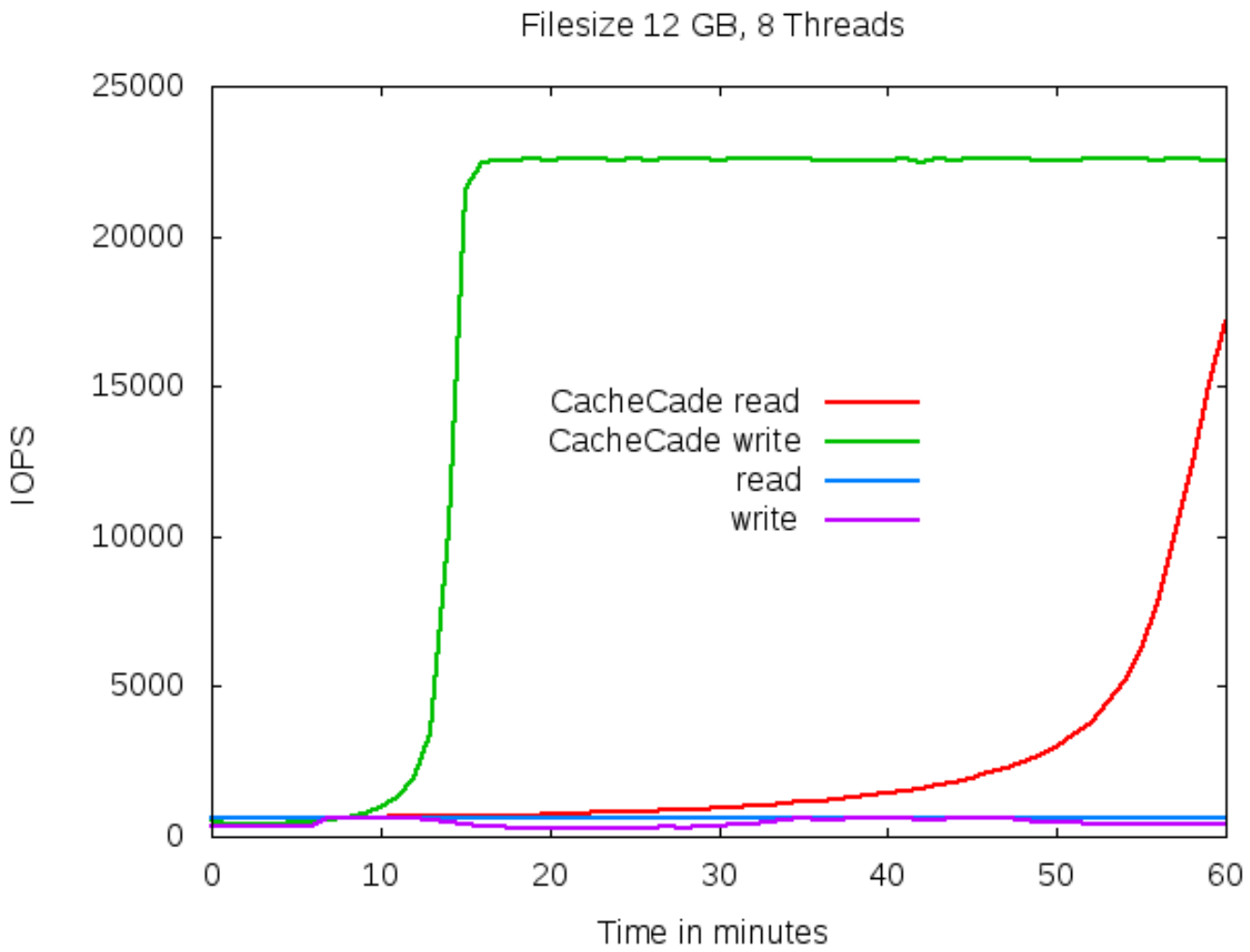



- Gut zu beobachten: Eine SSD hat sich verabschiedet!

# Zwischenstand CacheCade

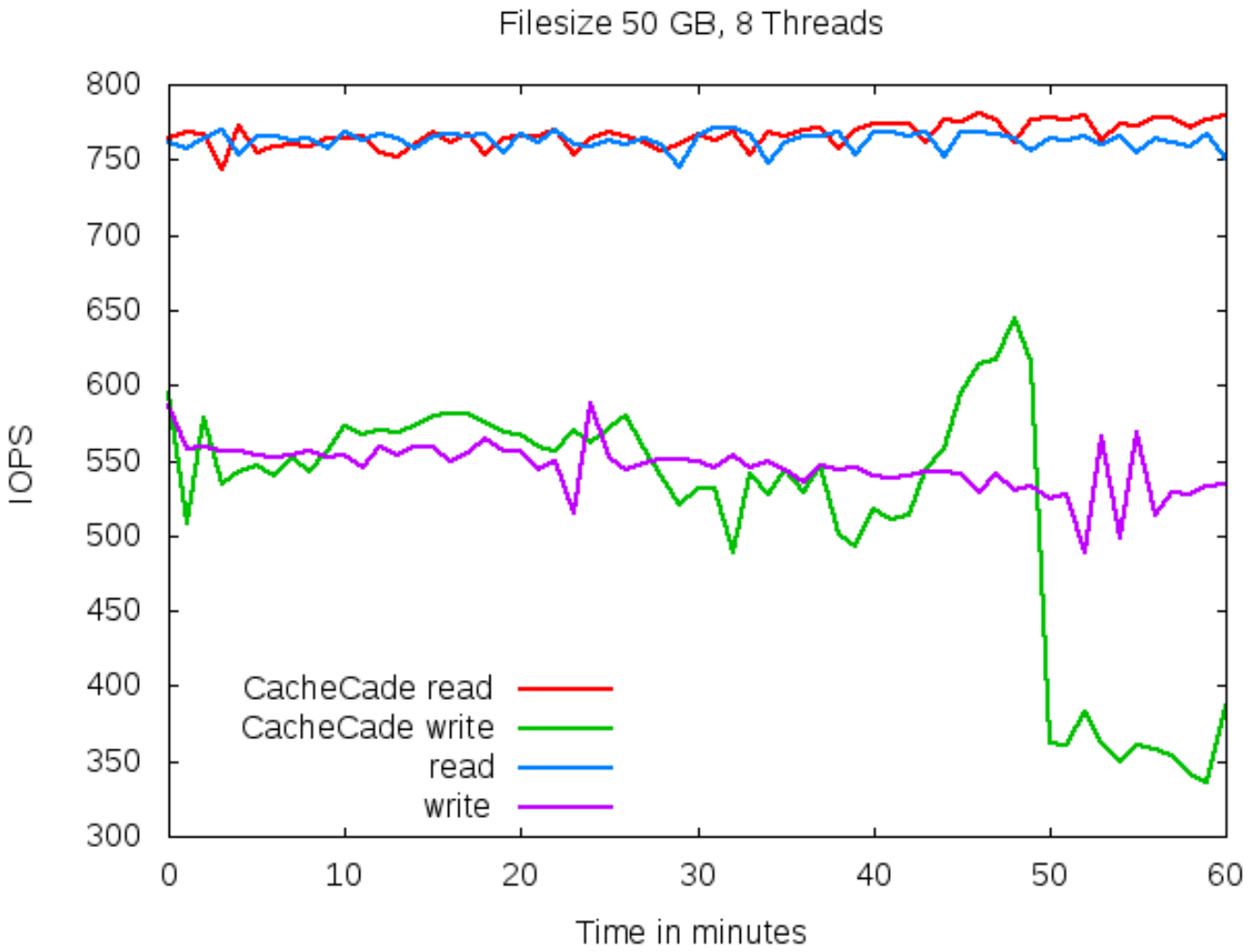
- Lernphase recht lang, dann deutlicher Effekt!
- Problem an Lastgrenze, Einbruch der Performance
- Ausfall einer SSD nicht akzeptabel
- OCZ-SSDs sind nicht für LSI-Controller zertifiziert
- OCZ sind Consumer-Disks.
- Test mit zertifizierten SSDs wiederholen -> INTEL.

# CacheCade mit Intel-SSDs (RAID 0)



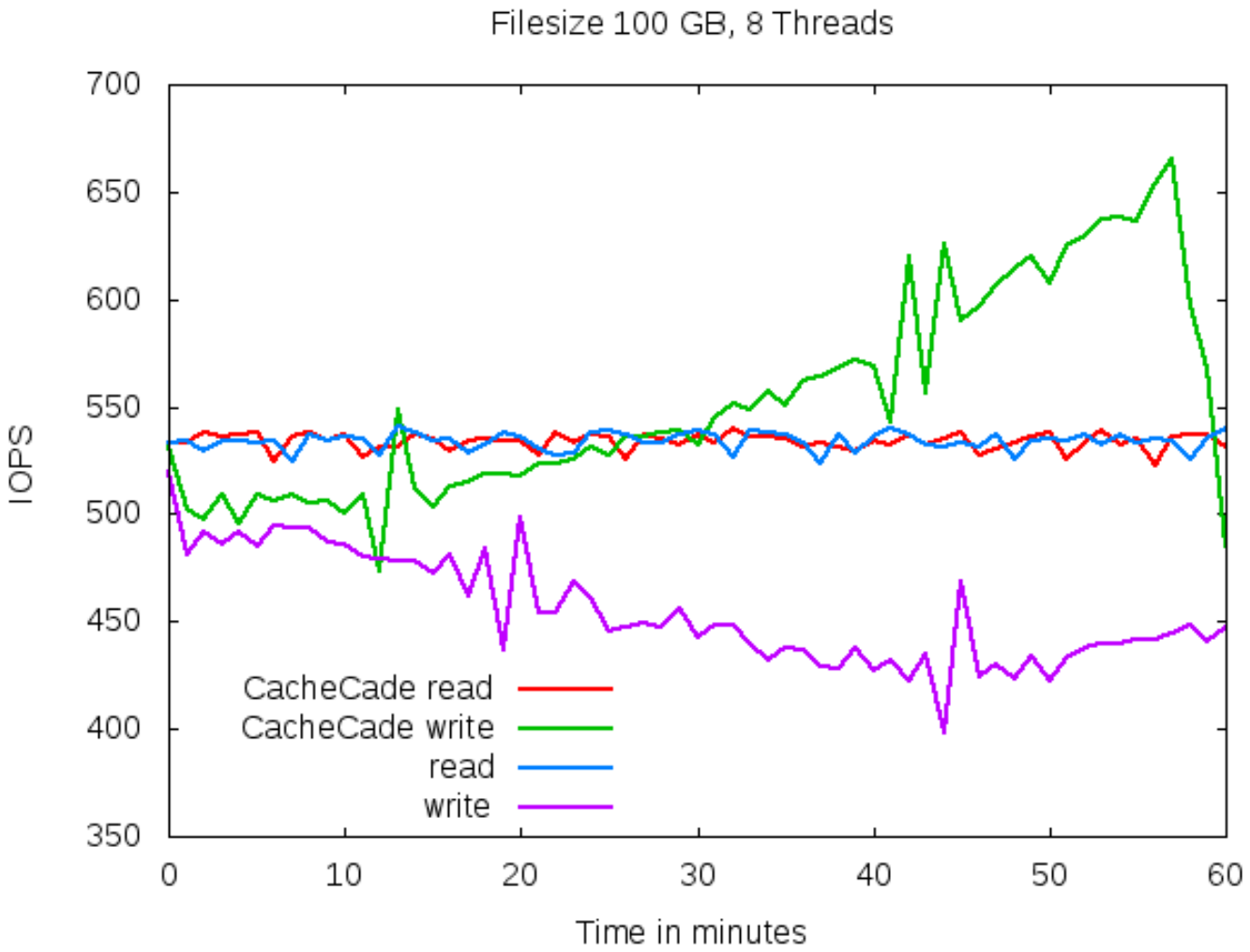

- Schreiben deutlich früher effektiv, Lesen braucht mehr Zeit

# CacheCade mit Intel-SSDs (RAID 0)




- kein wirklicher Effekt durch CacheCade zu erkennen!

# CacheCade mit Intel-SSDs (RAID 0)




- kein wirklicher Effekt durch CacheCade zu erkennen!

# Fazit CacheCade

- deutlicher Effekt bei kleinen Dateien
- Lernphase ist störend
- keine Wirkung bei Dateien größer dem CacheCade
- kleine Cache-Größe: Buffer/Cache des OS vermutlich effektiver!

$\Longrightarrow$ derzeit das Geld nicht wert!

# BCache

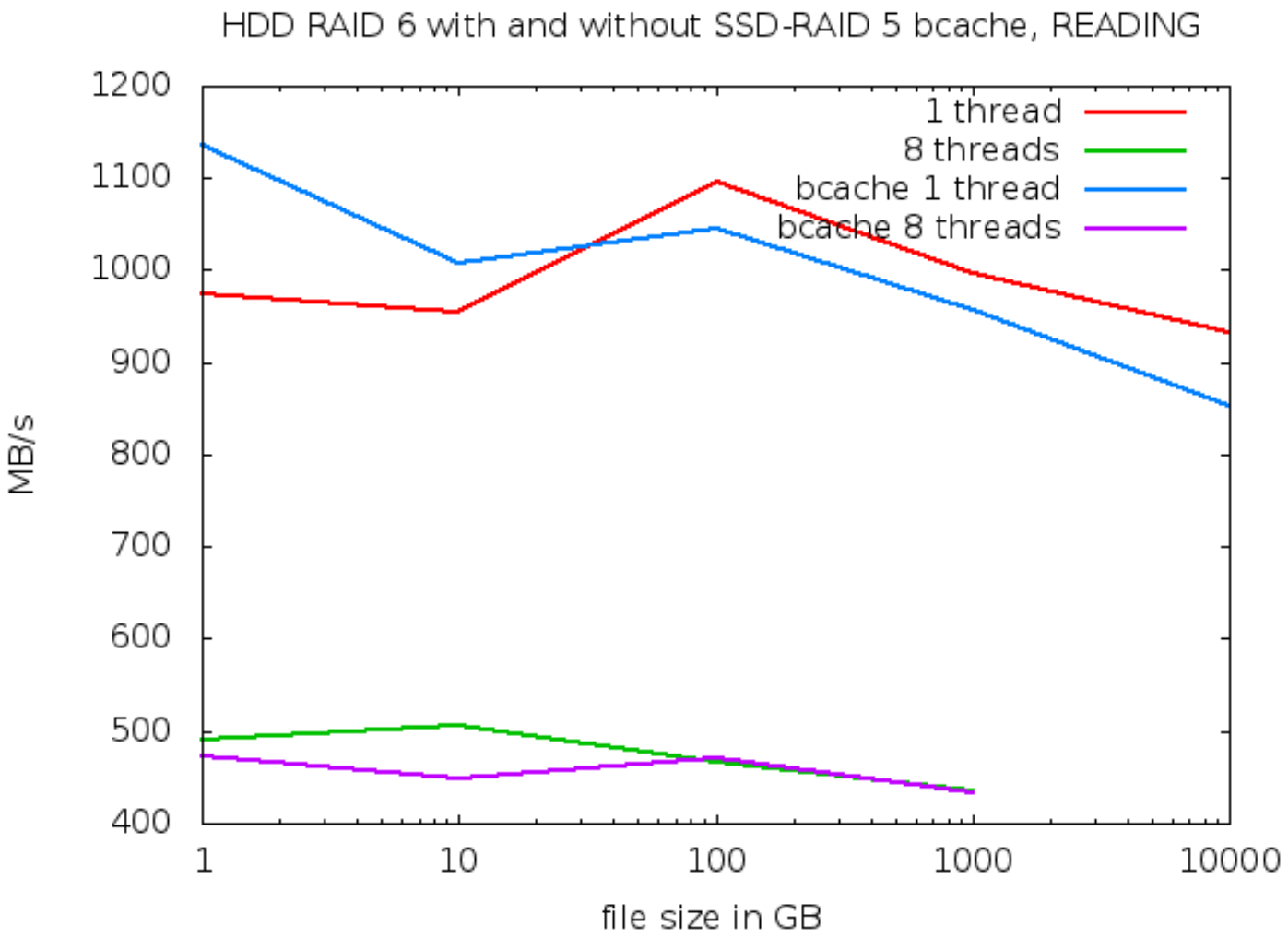
- Open Source, Bestandteil des Kernels ab 3.10
- keine Größenbeschränkung beim Cache
- keine Einschränkungen bei RAID-Verwendung
- keine Lernphase notwendig
- per default nur WriteThrough (Ausfall einzelner SSD kritisch)
- vielfältig anpassbar
- kein SSD-Caching bei sequentiellen Zugriffen:
  - hilft nicht, HDDs sind hier schon schnell
  - würde Cache-Daten invalidieren

$\Longrightarrow$ klingt vielversprechend!

# Vorgehen beim Testen

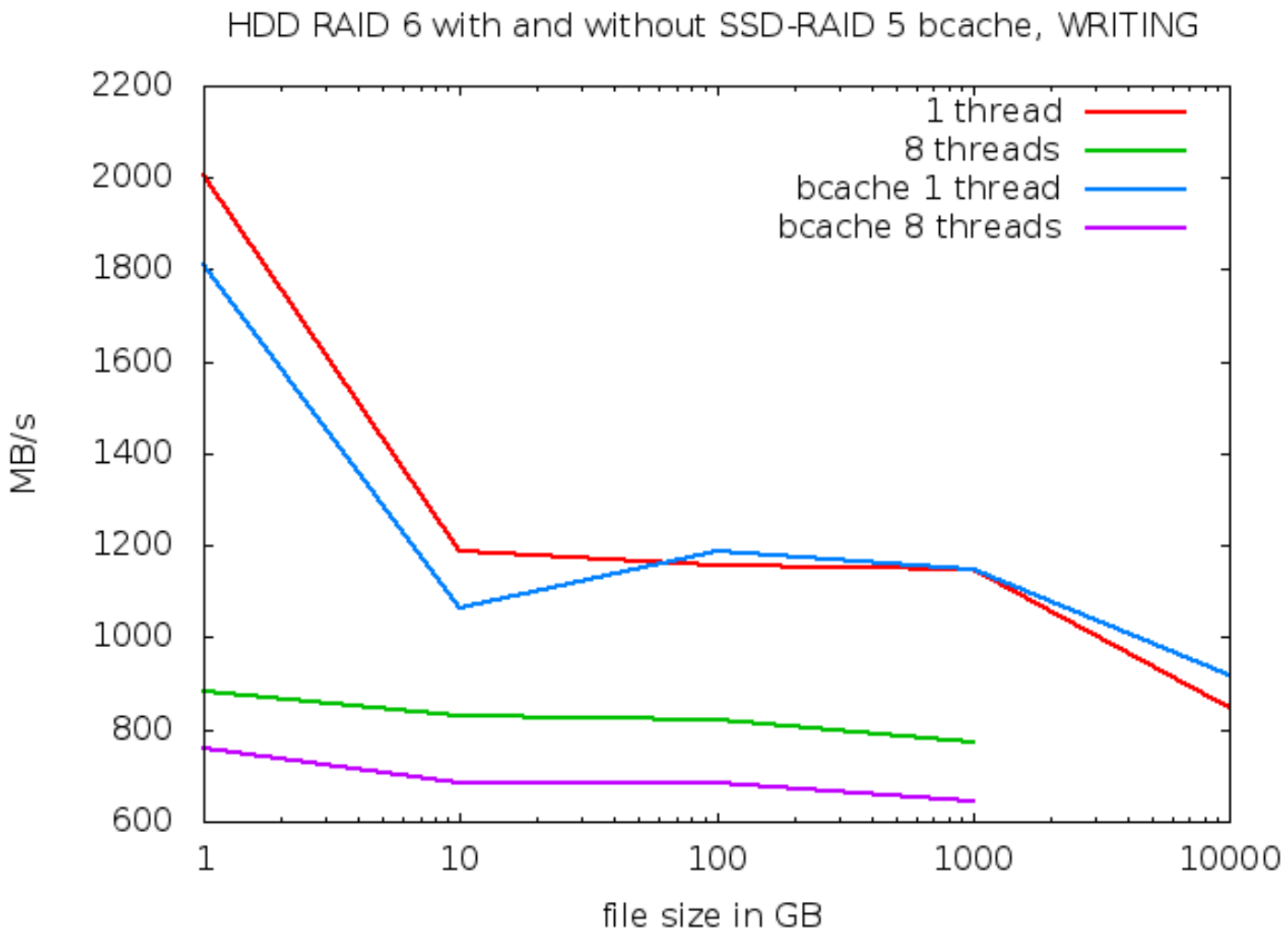
- Sicherstellen, dass sequentielle Zugriffe nicht langsamer sind
- Tests mit verschiedenen Dateigrößen: 1 GB, 10 GB, 100 GB, 1.000 GB, 10.000GB
- Random IOPS 60 Sekunden, danach gleiche Anzahl an Blöcken für sequentielle IOPS
- 60 Messungen in Folge mit 1 Thread und 8 Threads, lesend und schreibend
- IOPS-Vergleich mit 4k Blöcken von
  1. SSDs als RAID 0
  2. SSDs als RAID 5
  3. HDDs als RAID 6 ohne BCache
  4. HDD-RAID 6 mit SSD-RAID 0 als BCache
  5. HDD-RAID 6 mit SSD-RAID 5 als BCache
- rund 20.000 Messungen!

# Vergleich BCache seriell lesend



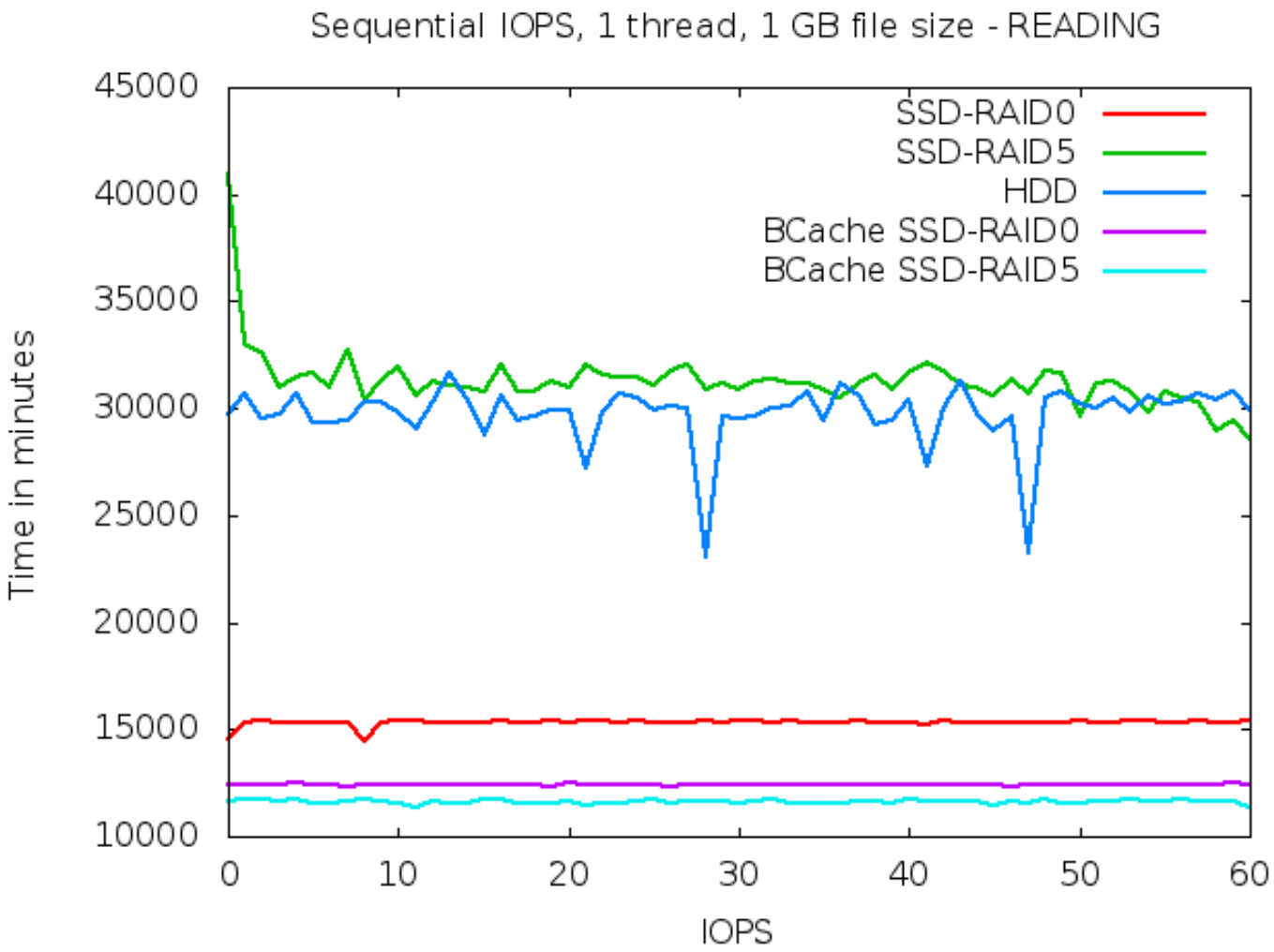

- kein wirklicher Effekt durch BCade zu erkennen! :-)

# Vergleich BCache seriell schreibend



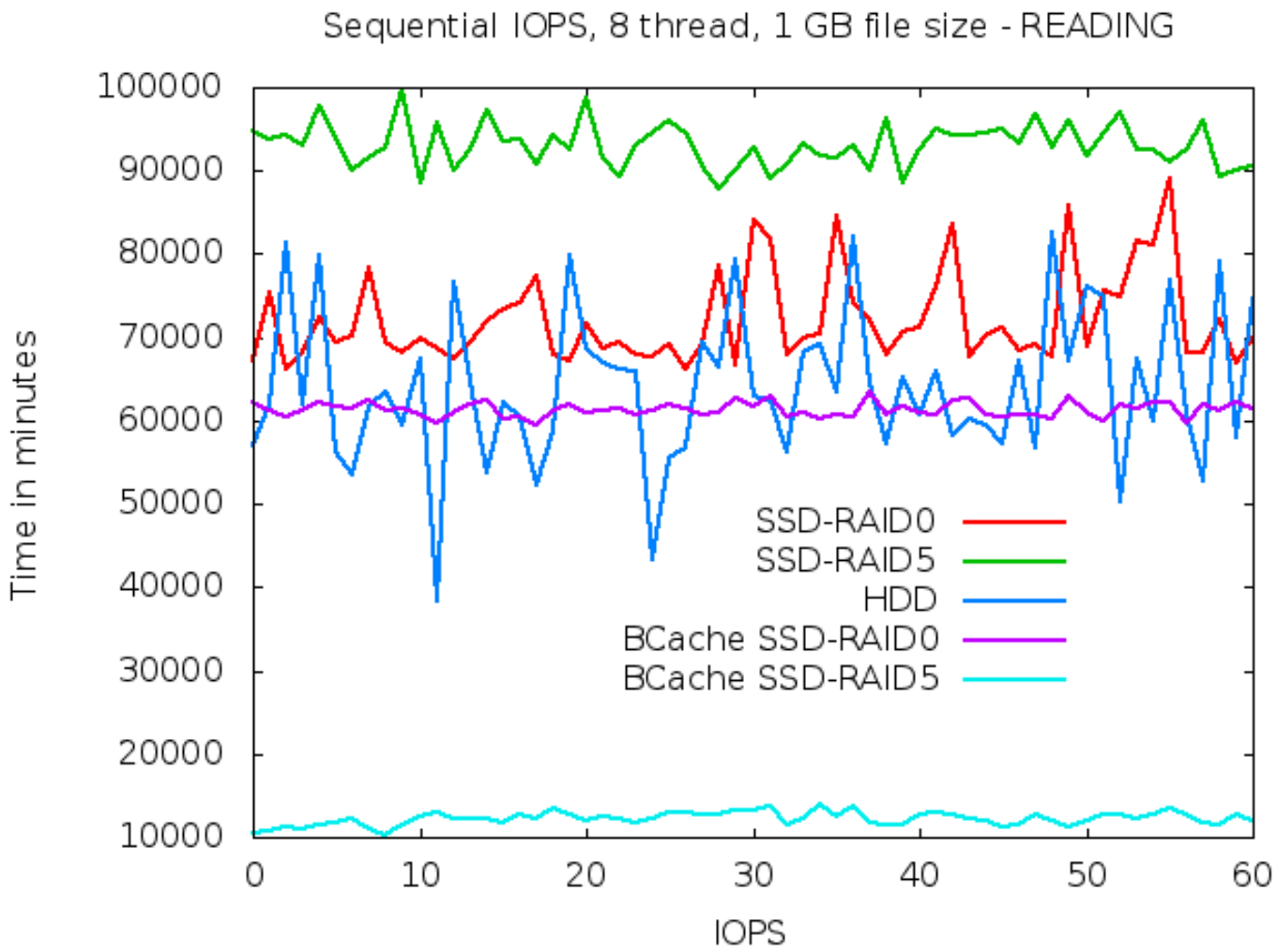

- kein wirklicher Effekt durch BCade zu erkennen! :-)

# Vergleich BCache-IOPS seriell lesend, 1 GB



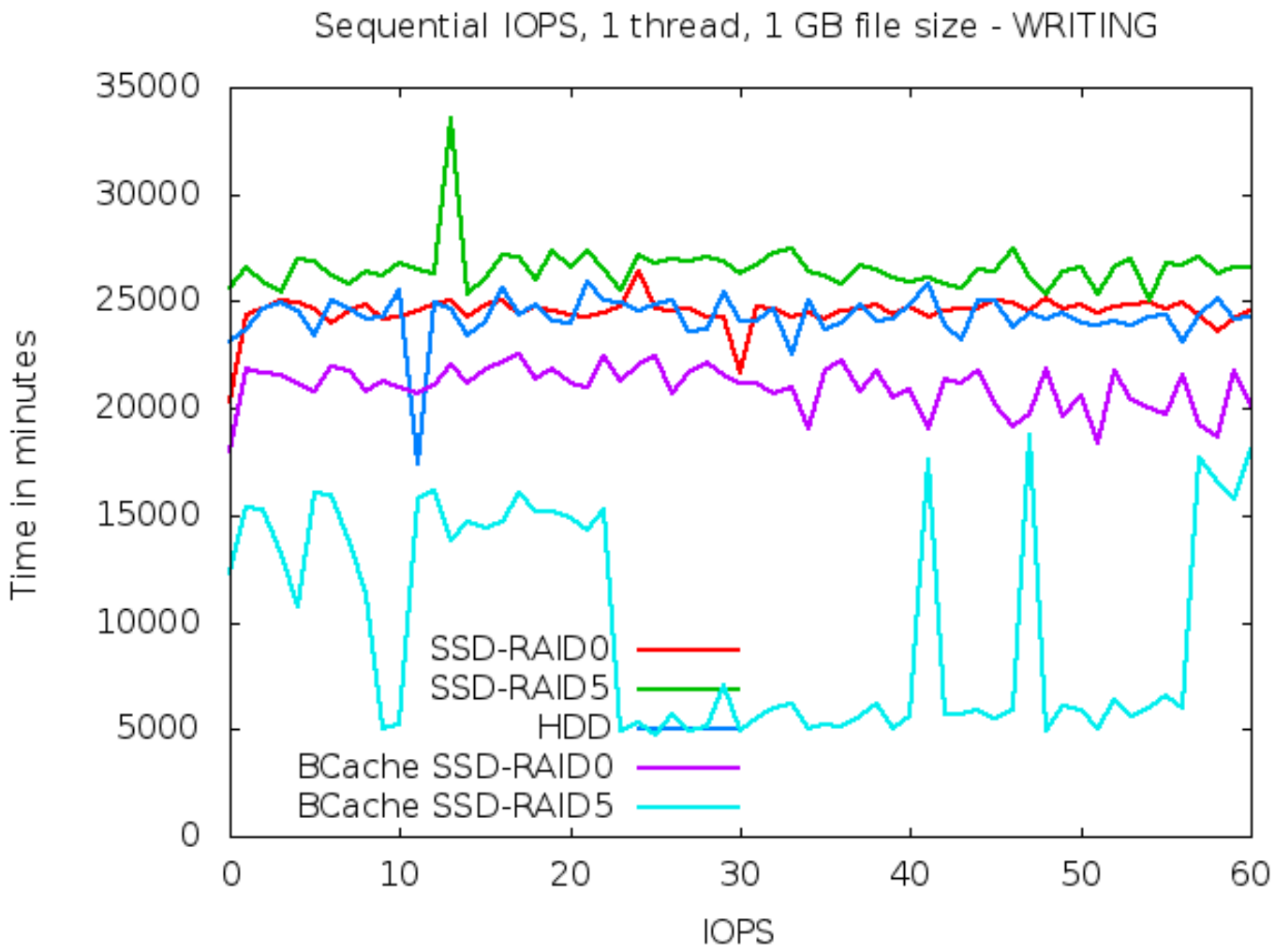

- SSD RAID 5 schneller wegen Striping
- HDD RAID 6 schneller wegen read ahead

# Vergleich BCache-IOPS seriell lesend, 1 GB



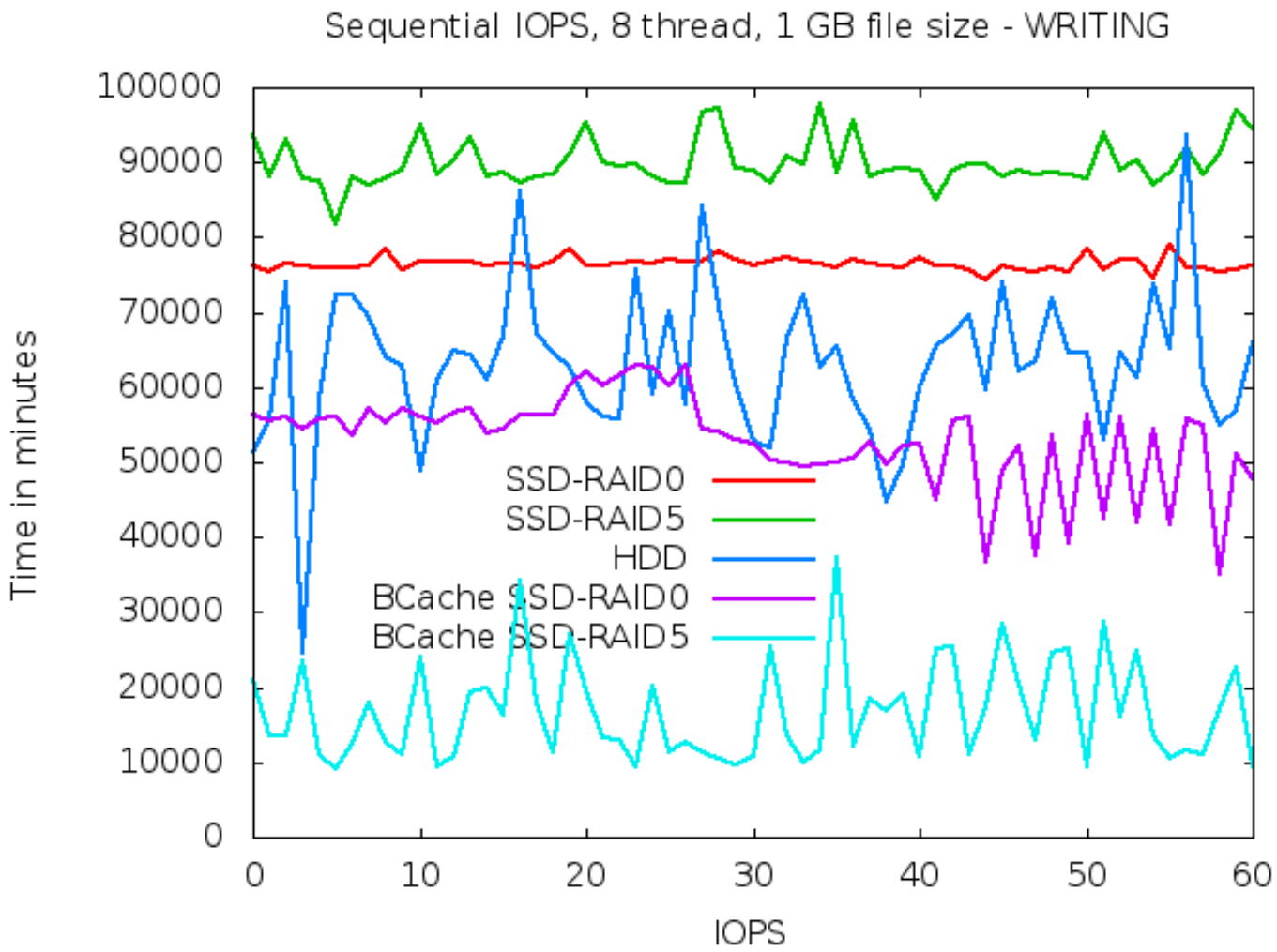

- SSD RAID 5 schneller wegen Striping
- BCache SSD RAID 5 vermutlich wegen doppeltem RAID 5/6

# Vergleich BCache-IOPS seriell schreibend, 1 GB



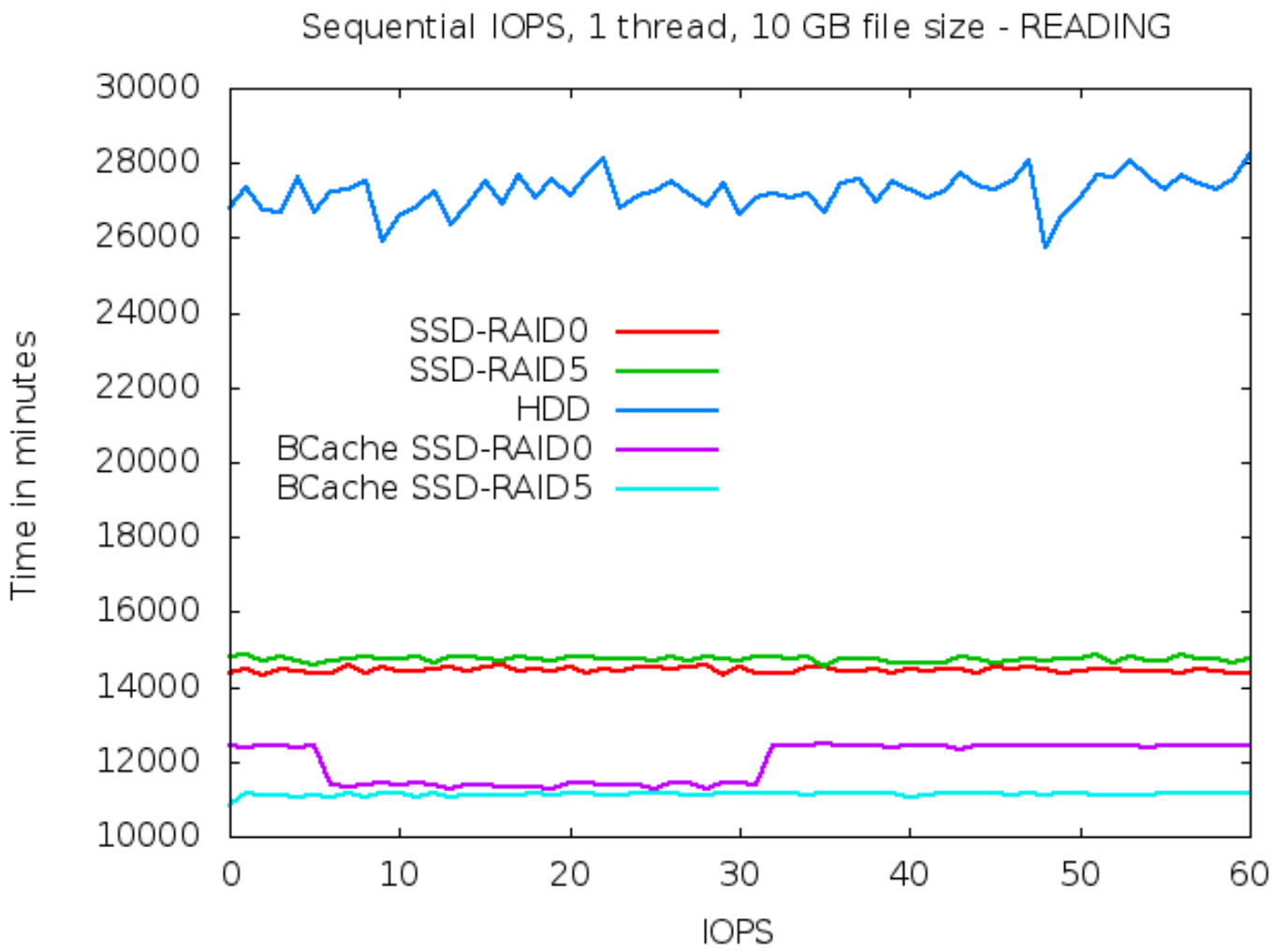

- BCache mit SSD RAID 5 seltsam, vermutlich Parity-Berechnung
- RAM-Cache des RAID-Controllers spielt eine Rolle!

# Vergleich BCache-IOPS seriell schreibend, 1 GB



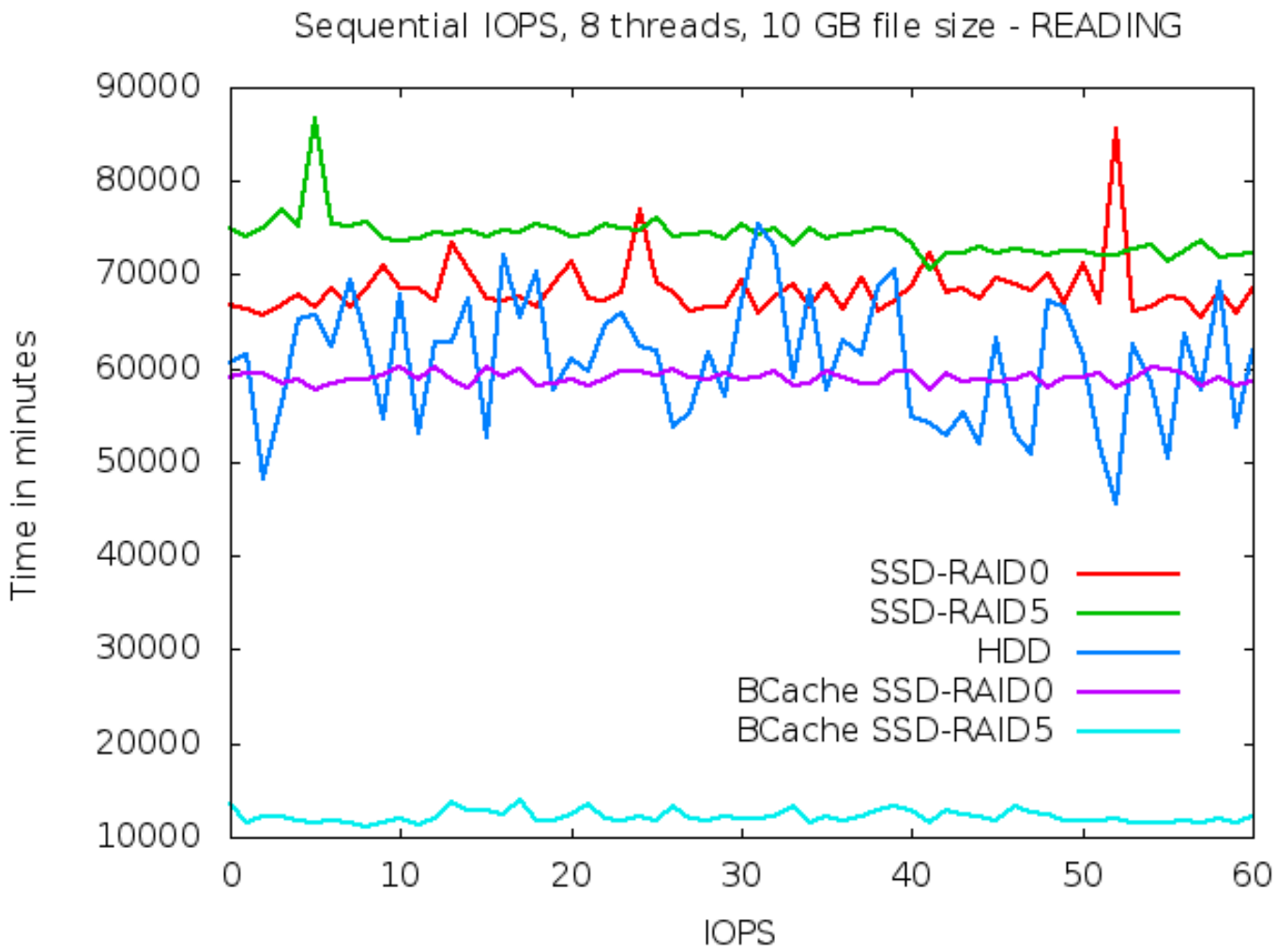

- BCache mit SSD RAID 5 vermutlich wegen doppeltem RAID 5/6
- RAM-Cache des RAID-Controllers spielt eine Rolle!

# Vergleich BCache-IOPS seriell lesend, 10 GB



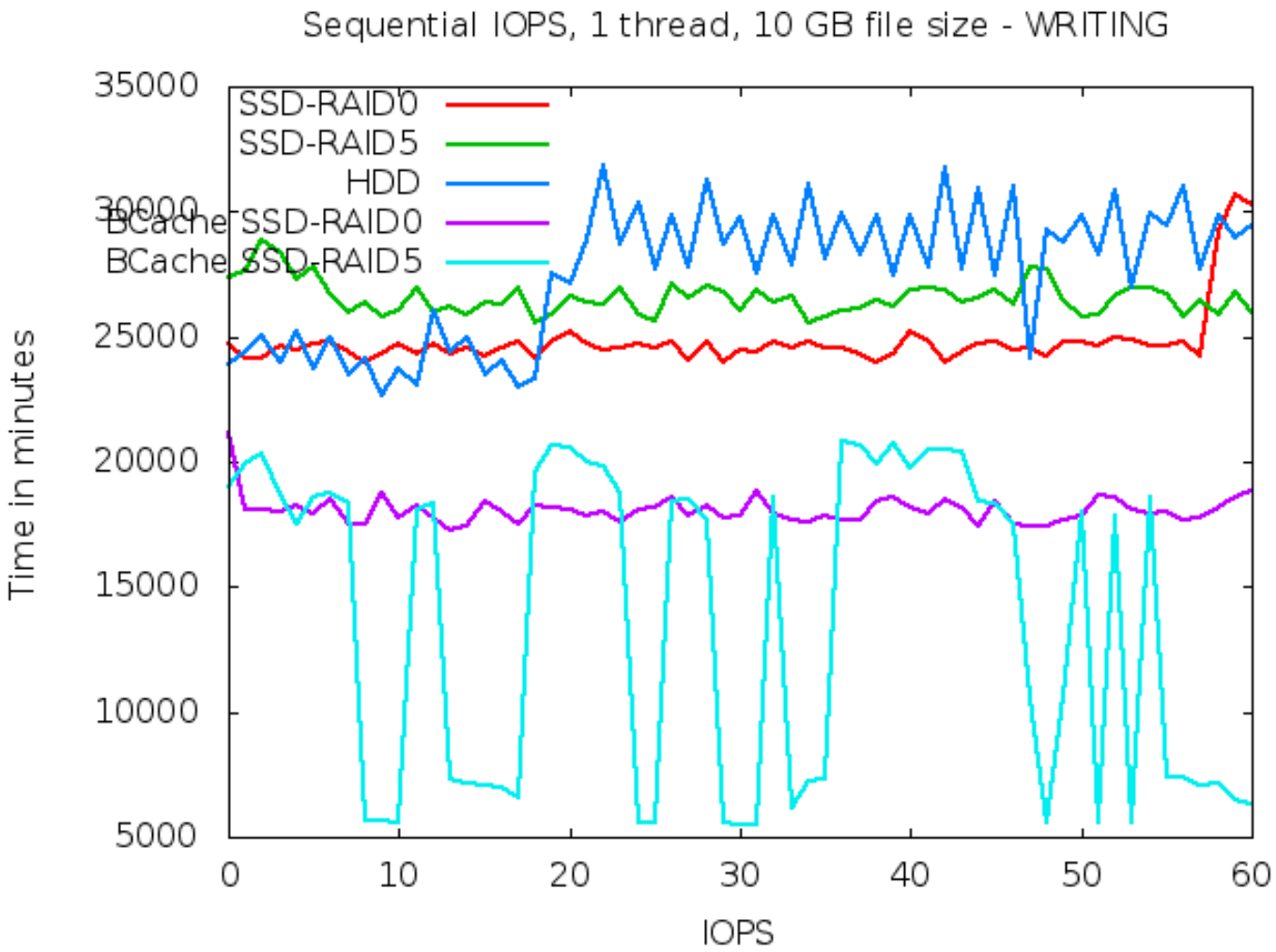

- HDD RAID 6 schneller wegen read ahead
- RAM-Cache des RAID-Controller verliert Einfluss

# Vergleich BCache-IOPS seriell lesend, 10 GB



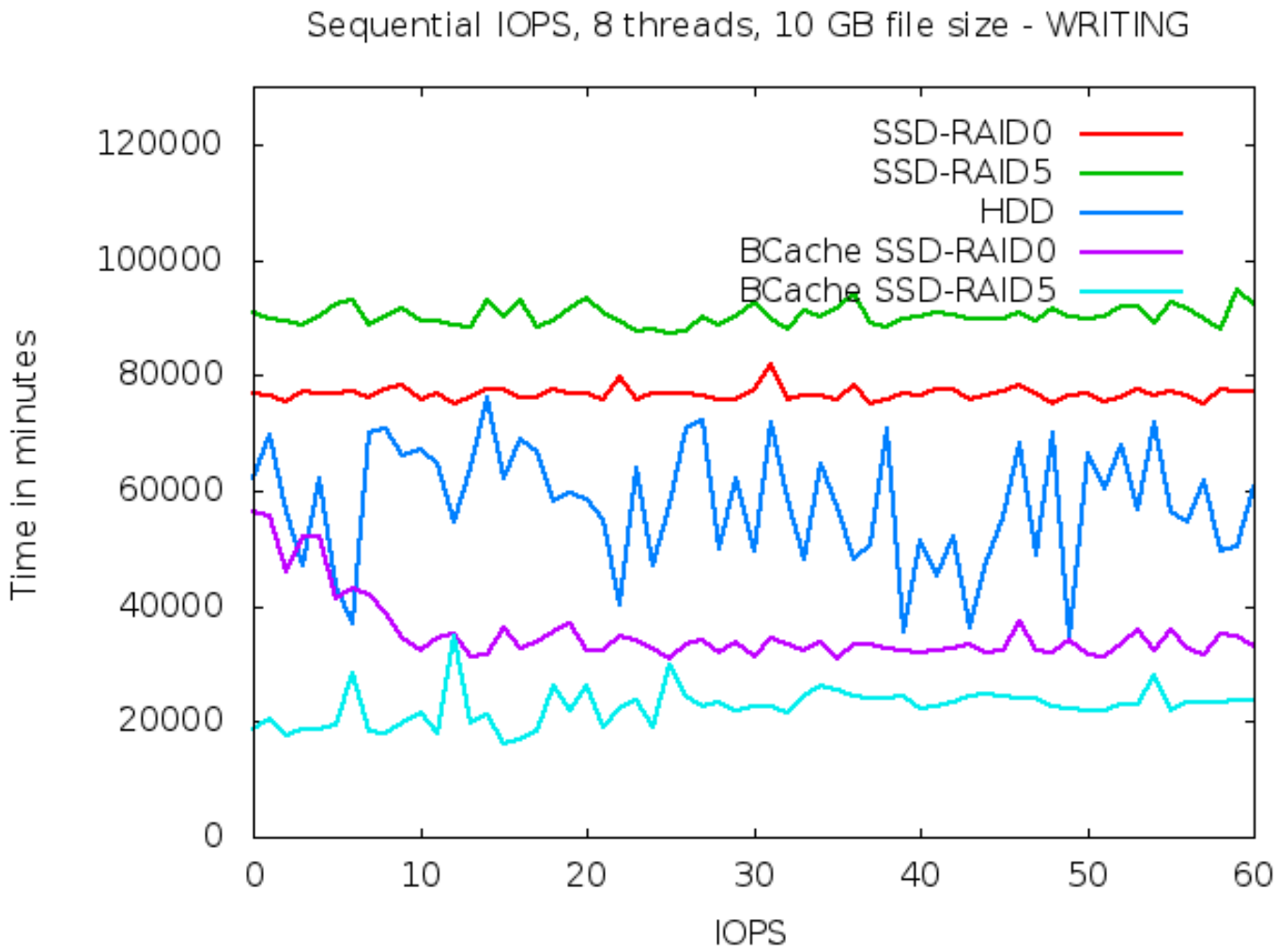

- mehrere Threads: SSDs zeigen Latenzgewinn
- BCache SSD RAID 5 vermutlich wegen doppeltem RAID 5/6

# Vergleich BCache-IOPS seriell schreibend, 10 GB




- BCache mit SSD RAID 5 seltsam, vermutlich Parity-Berechnung

# Vergleich BCache-IOPS seriell schreibend, 10 GB

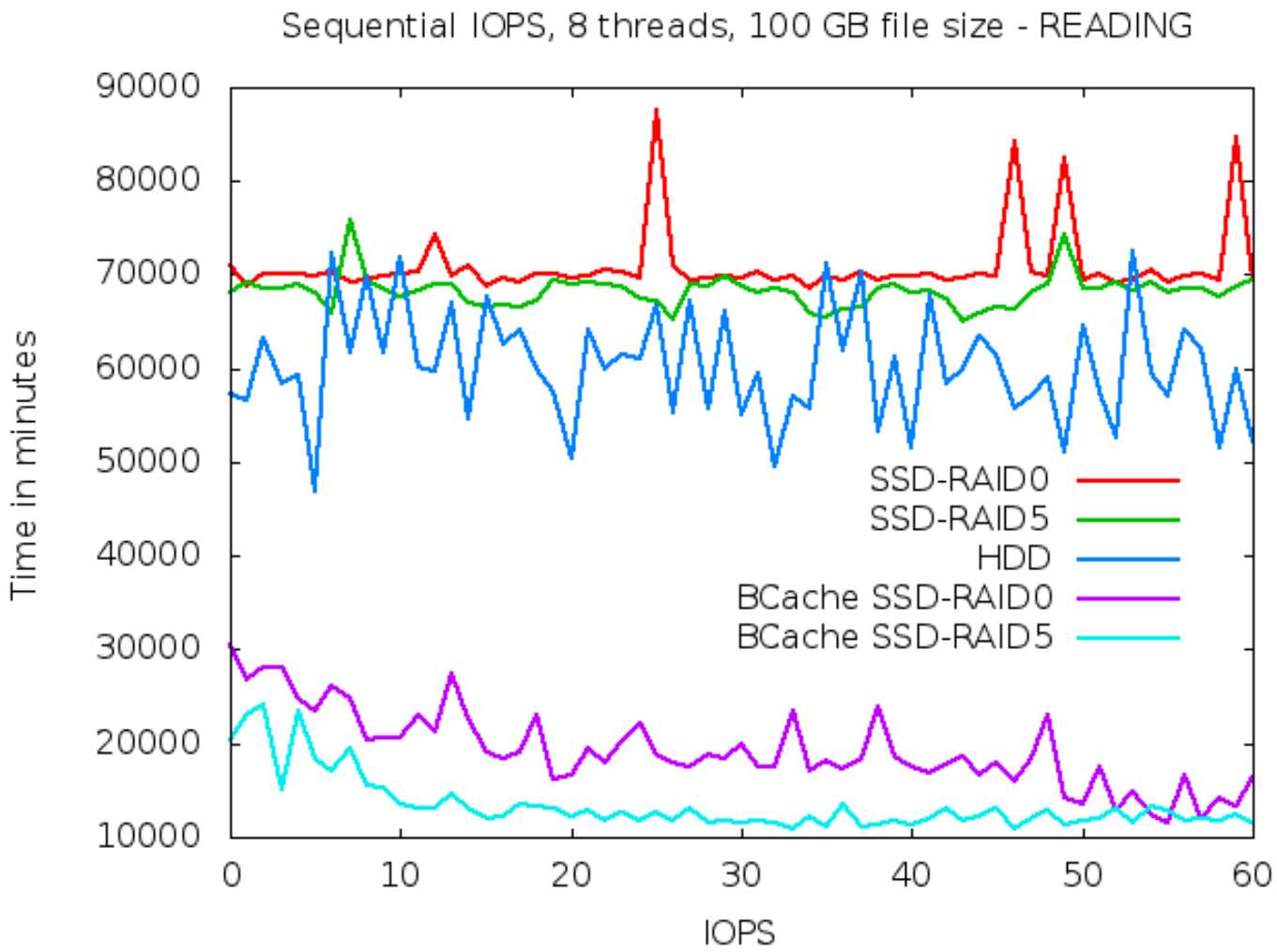



- BCache mit SSD RAID 5 vermutlich wegen doppeltem RAID 5/6

# Vergleich BCache-IOPS seriell lesend, 100 GB


- SSD RAID 5 schneller wegen Striping
- HDD RAID 6 schneller wegen read ahead

# Vergleich BCache-IOPS seriell lesend, 100 GB

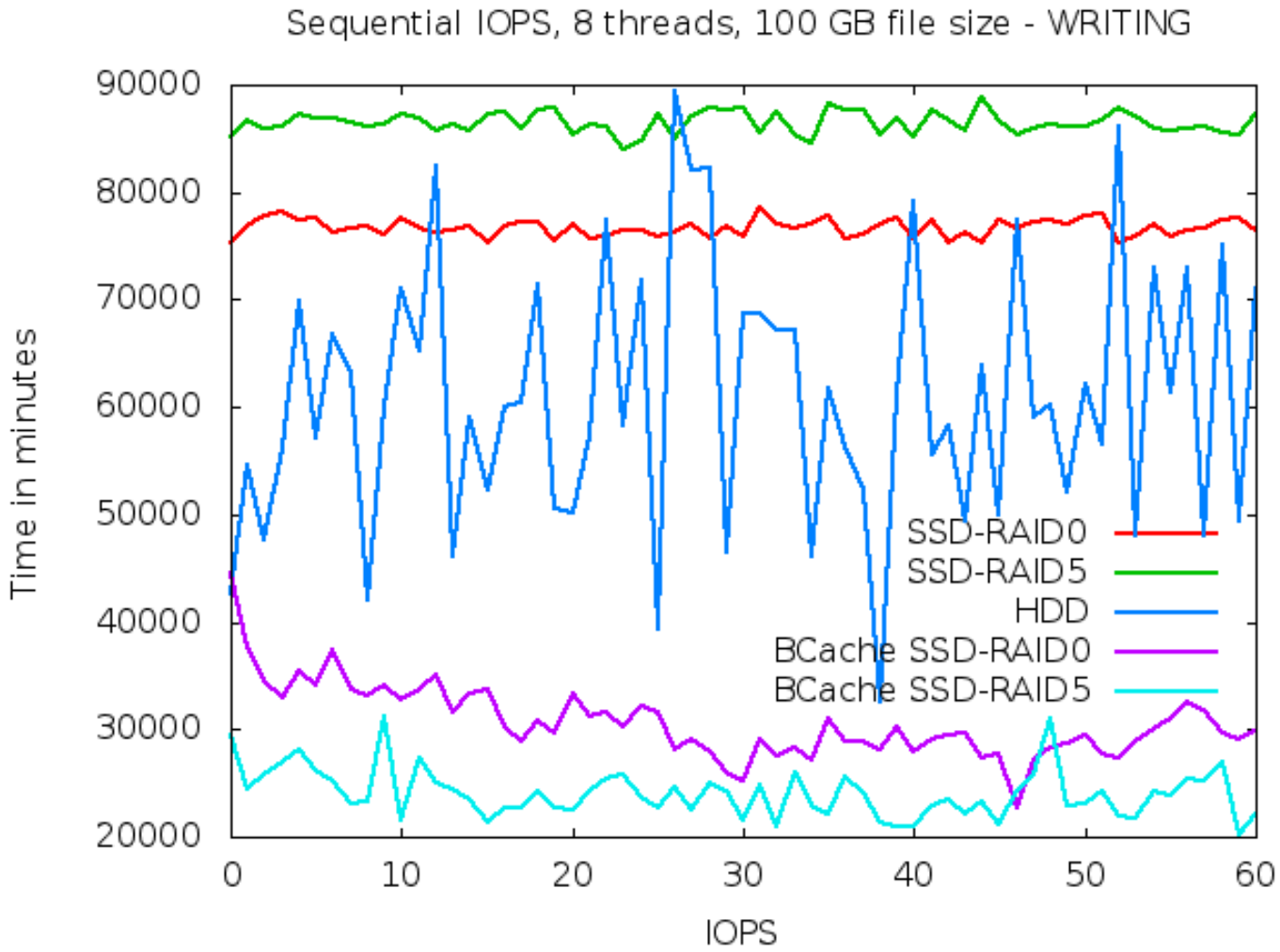



- mehrere Threads: Latenzgewinn der SSDs
- BCache SSD RAID 5 vermutlich wegen doppeltem RAID 5/6

# Vergleich BCache-IOPS seriell schreibend, 100 GB

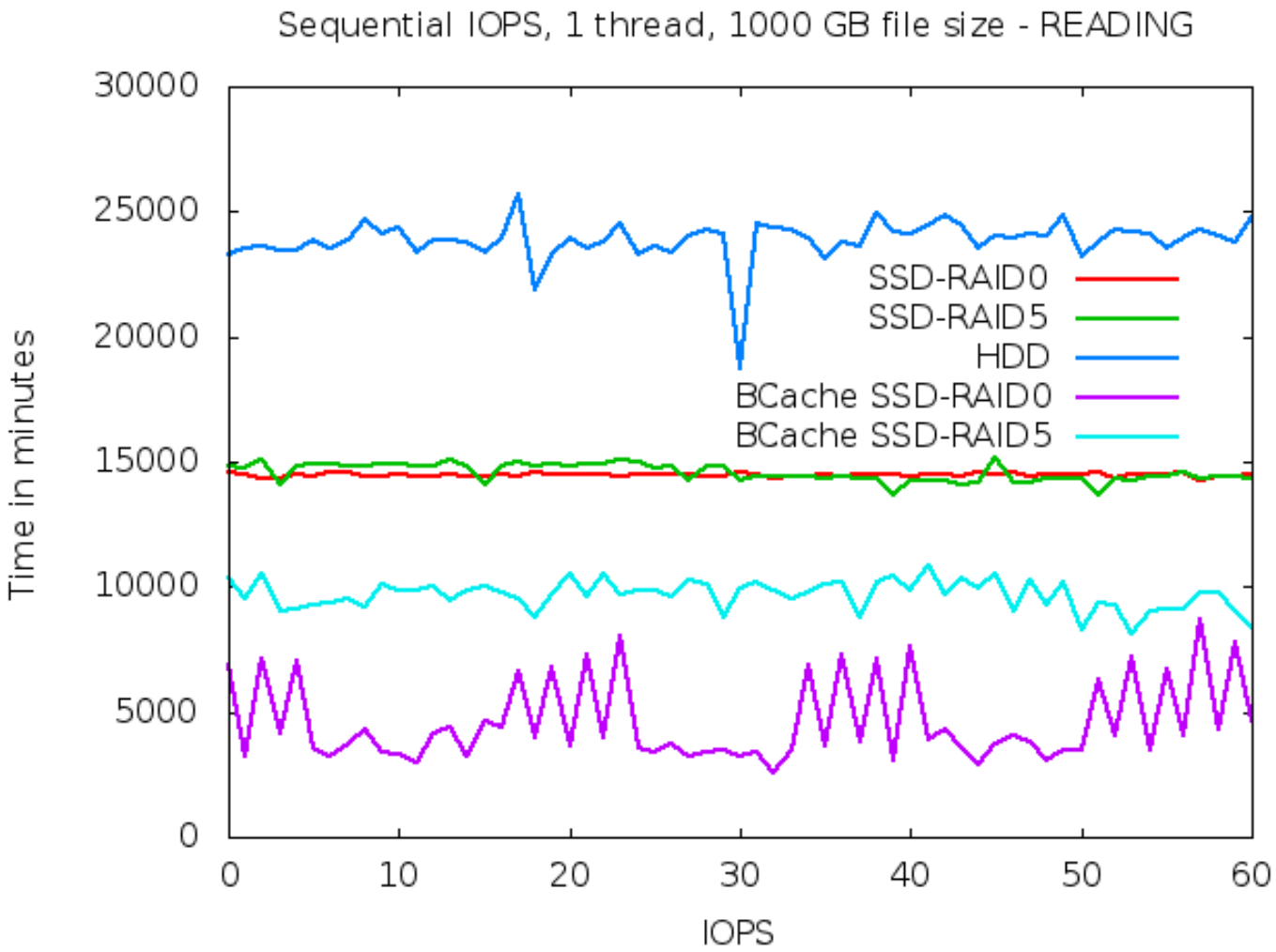

- BCache mit SSD RAID 5 seltsam, vermutlich Parity-Berechnung
- BCache verliert: sequentiell schreiben von 4k Blöcken ungewöhnlich!

# Vergleich BCache-IOPS seriell schreibend, 100 GB



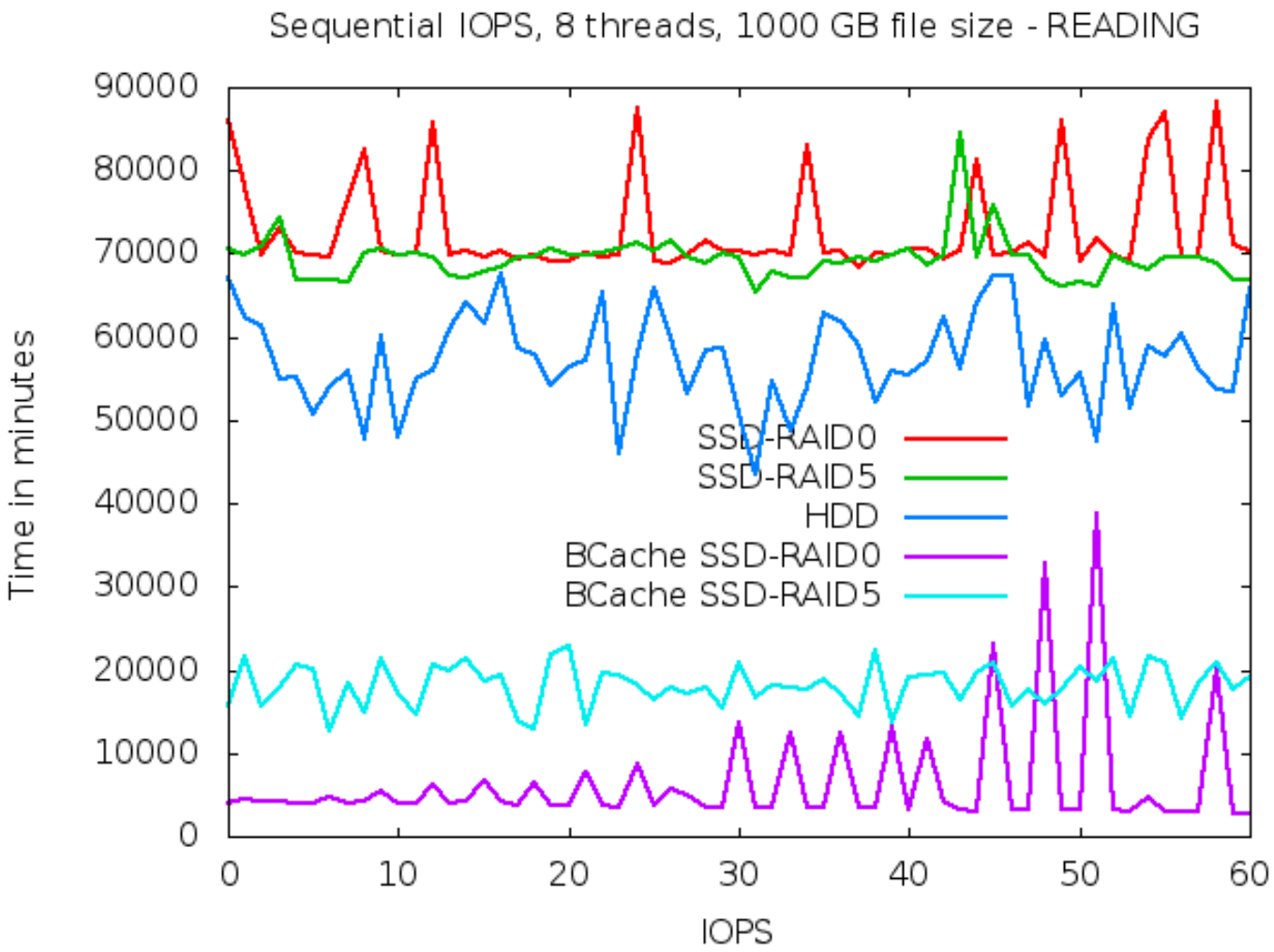

- BCache verliert: sequentiell schreiben von 4k Blöcken ungewöhnlich!

# Vergleich BCache-IOPS seriell lesend, 1000 GB



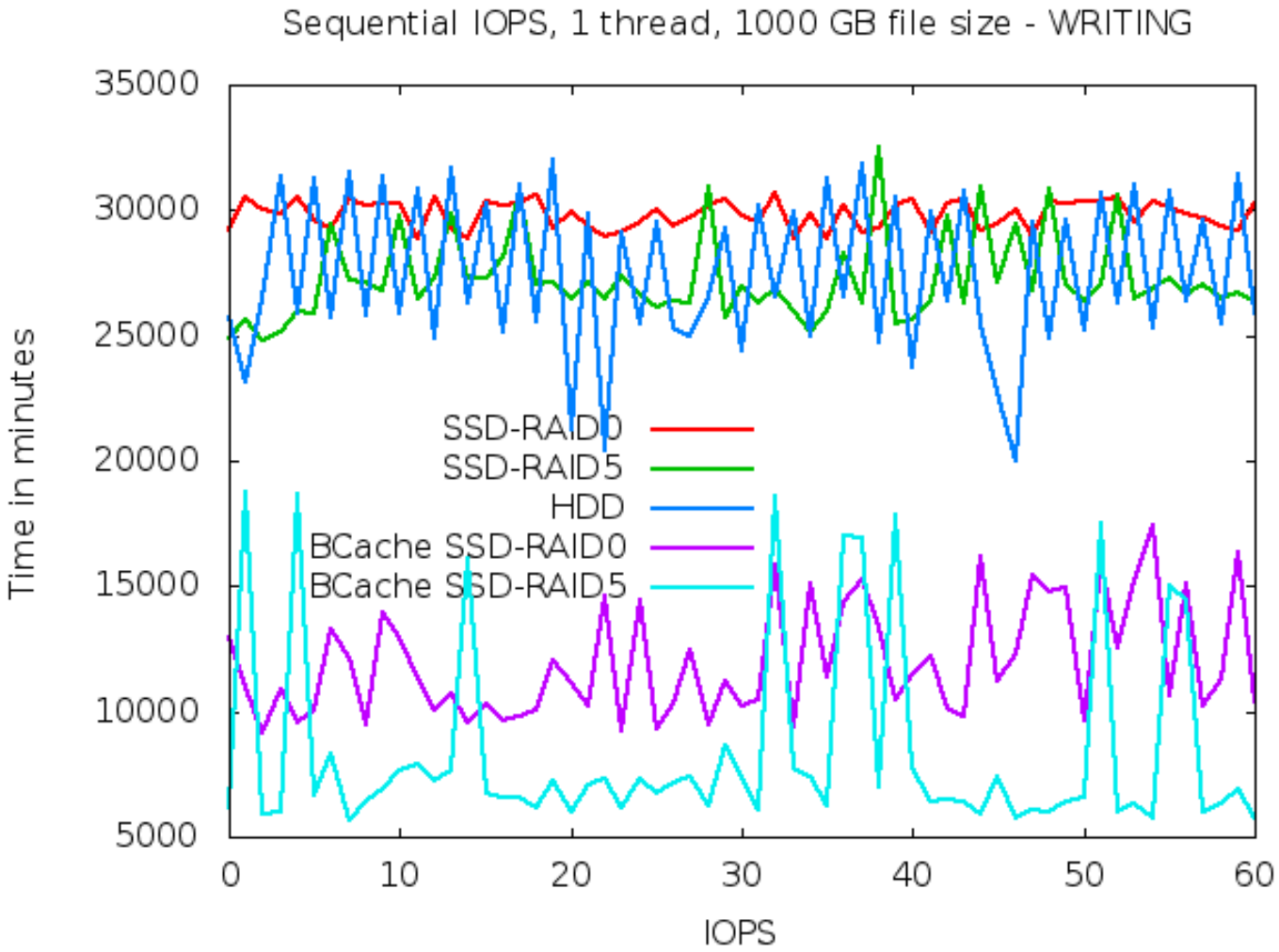

- HDD RAID 6 schneller wegen read ahead

# Vergleich BCache-IOPS seriell lesend, 1000 GB



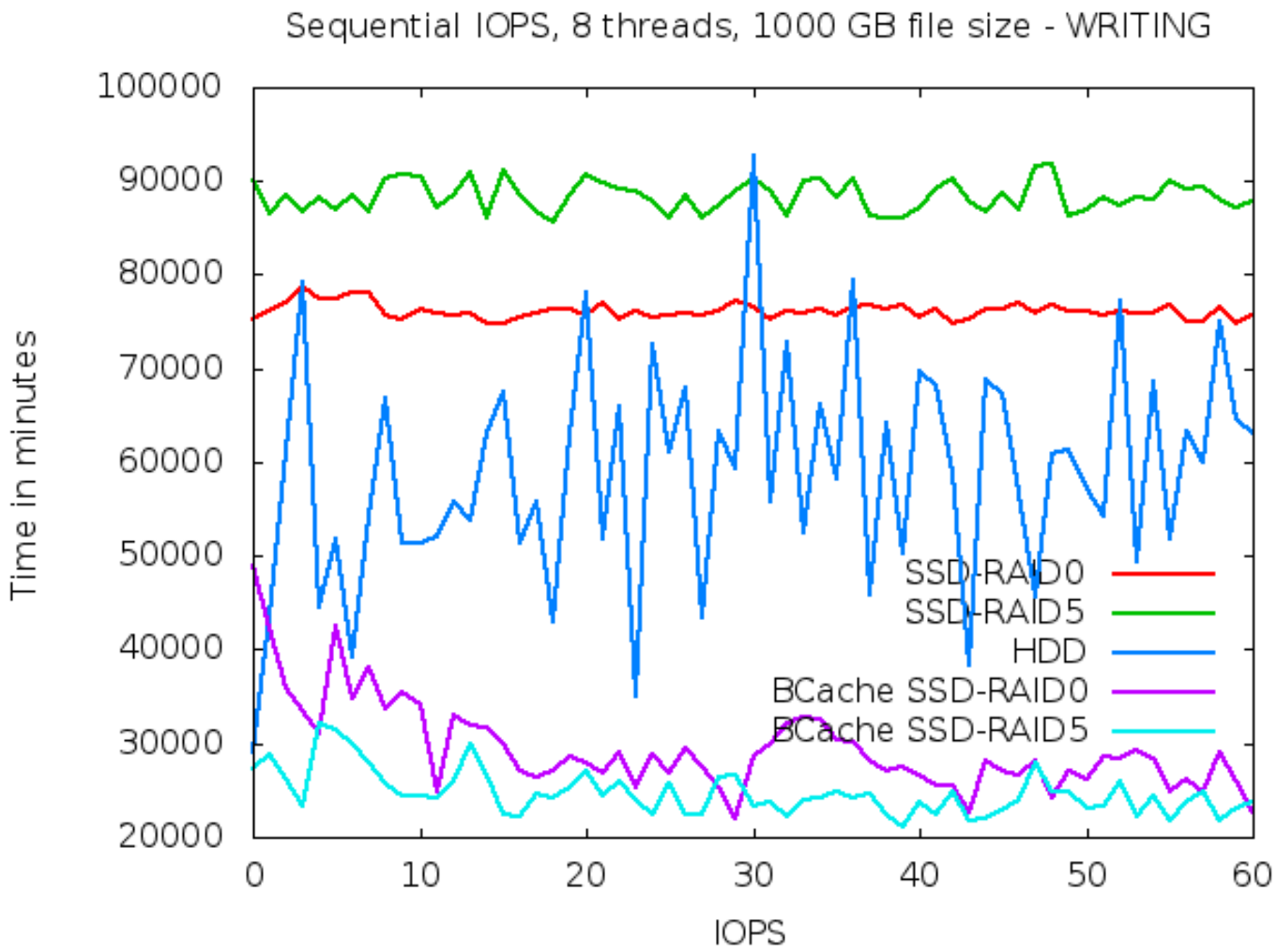

- mehrere Threads: Latenzgewinn der SSDs
- BCache SSD RAID 5 vermutlich wegen doppeltem RAID 5/6

# Vergleich BCache-IOPS seriell schreibend, 1000 GB



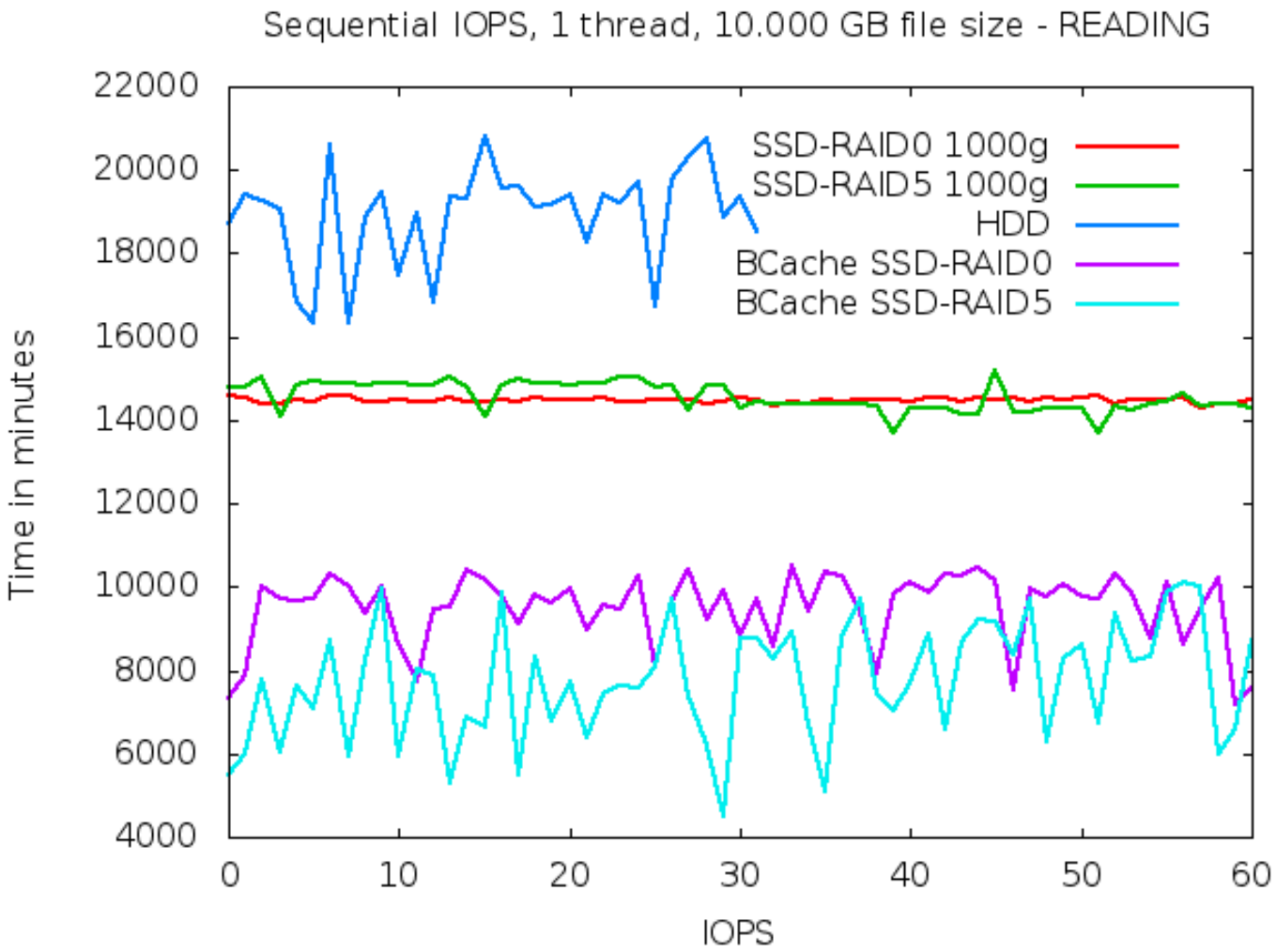

- BCache verliert: sequentiell schreiben von 4k Blöcken ungewöhnlich!

# Vergleich BCache-IOPS seriell schreibend, 1000 GB



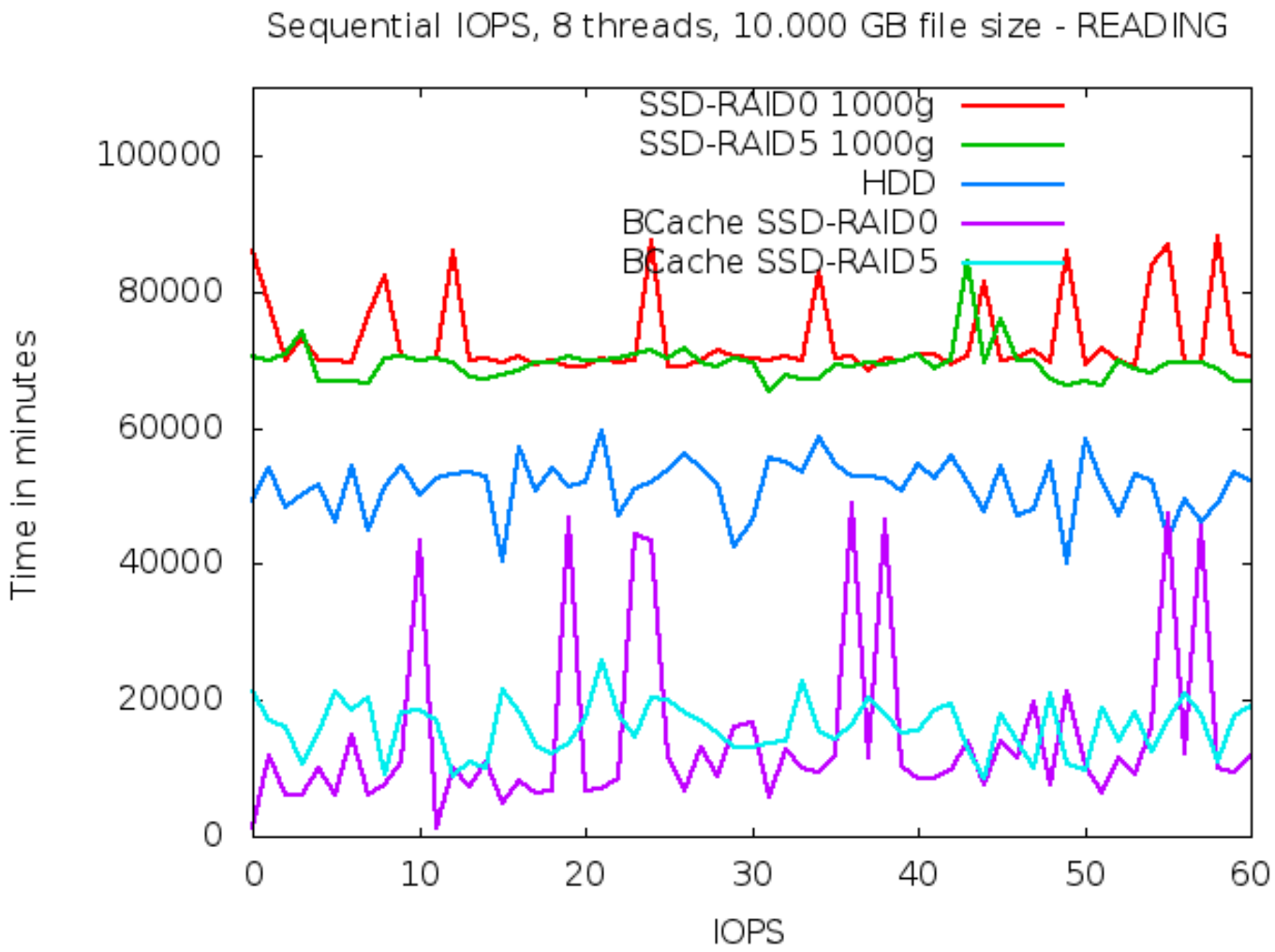

- BCache mit SSD RAID 5 vermutlich wegen doppeltem RAID 5/6
- mehrere Threads: Latenzgewinn der SSDs

# Vergleich BCache-IOPS seriell lesend, 10.000 GB



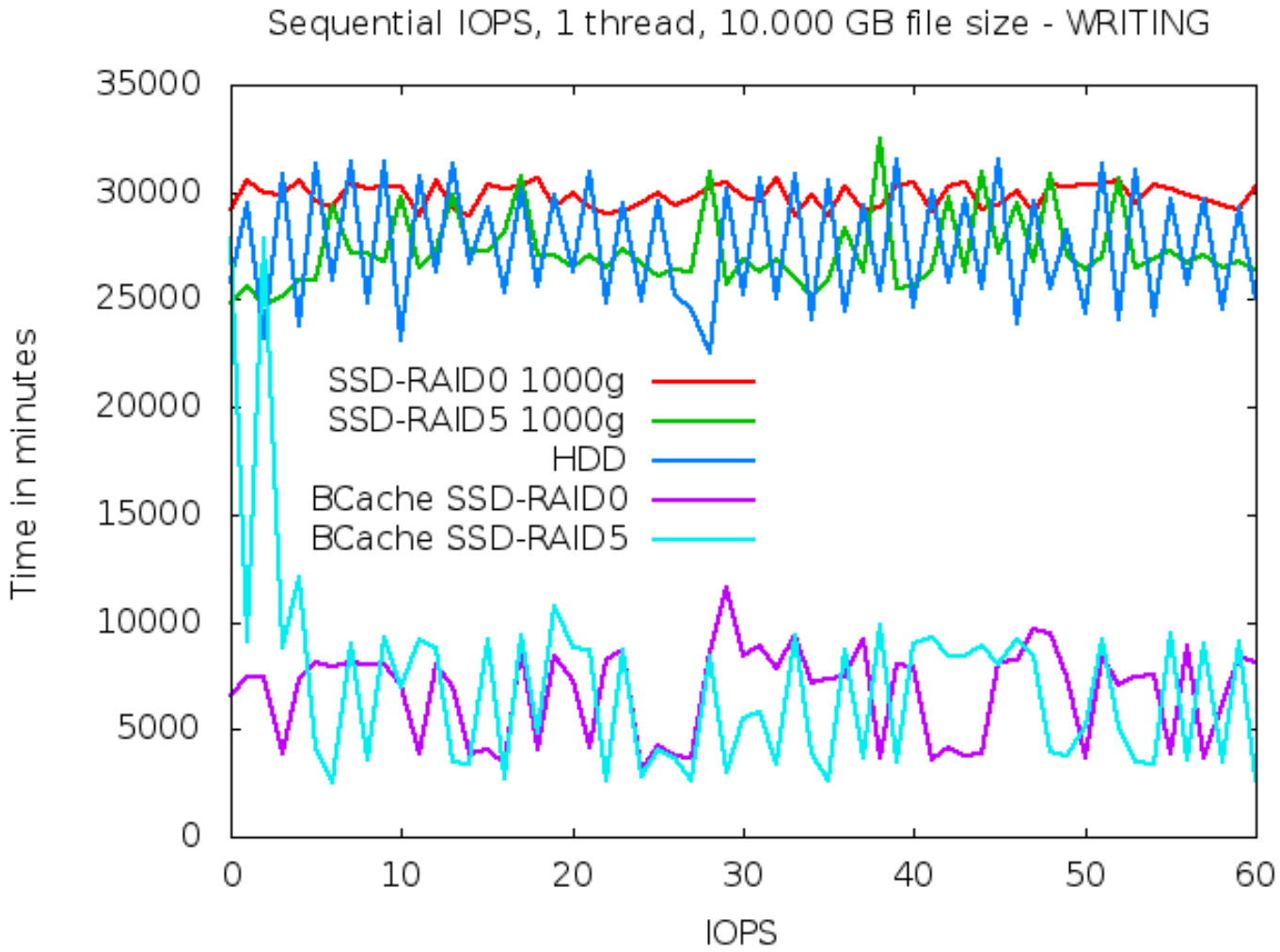

- HDD RAID 6 schneller wegen read ahead

# Vergleich BCache-IOPS seriell lesend, 10.000 GB



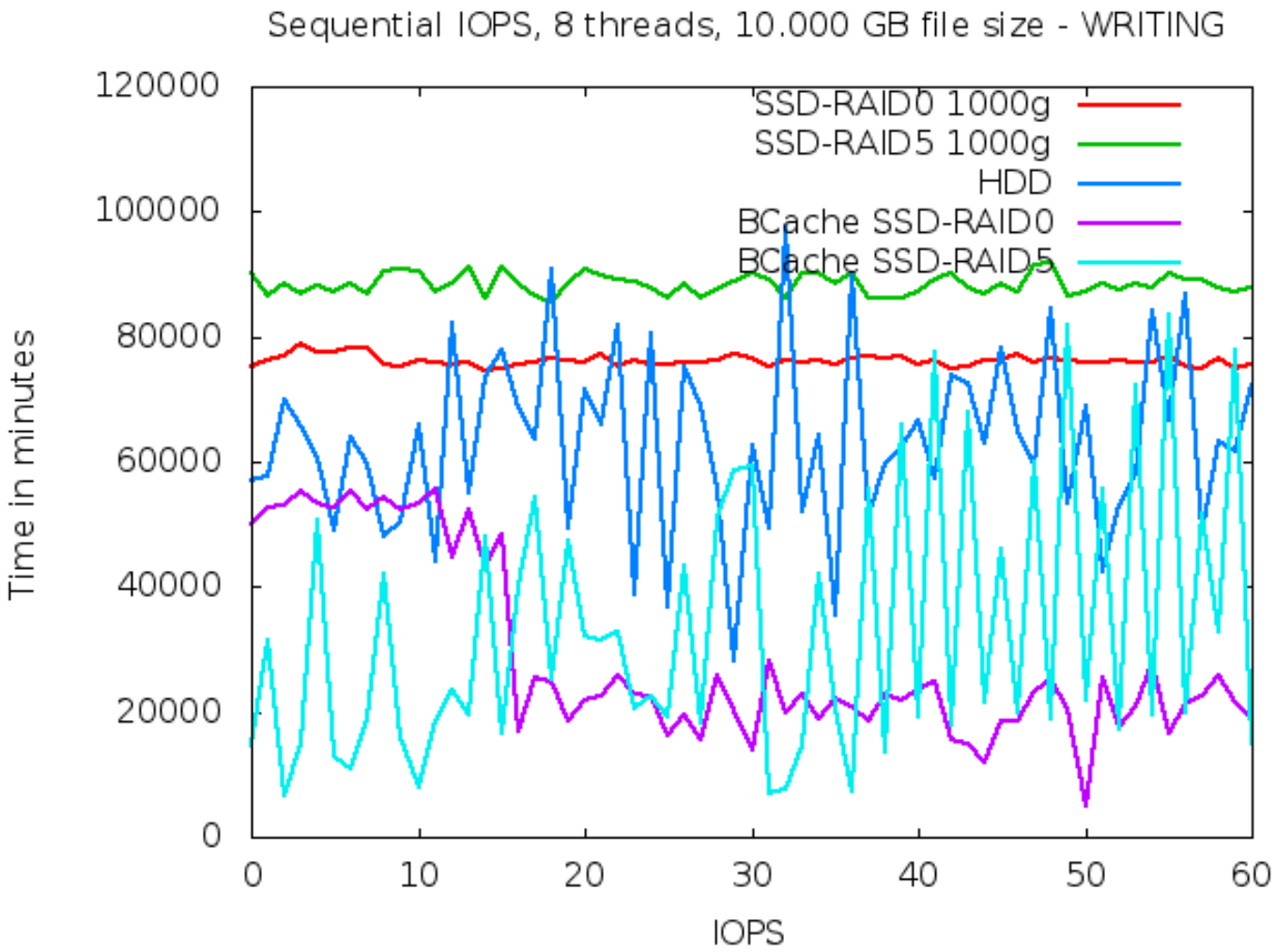

- mehrere Threads: Latenzgewinn der SSDs
- BCache SSD RAID 5 vermutlich wegen doppeltem RAID 5/6

# Vergleich BCache-IOPS seriell schreibend, 10.000 GB



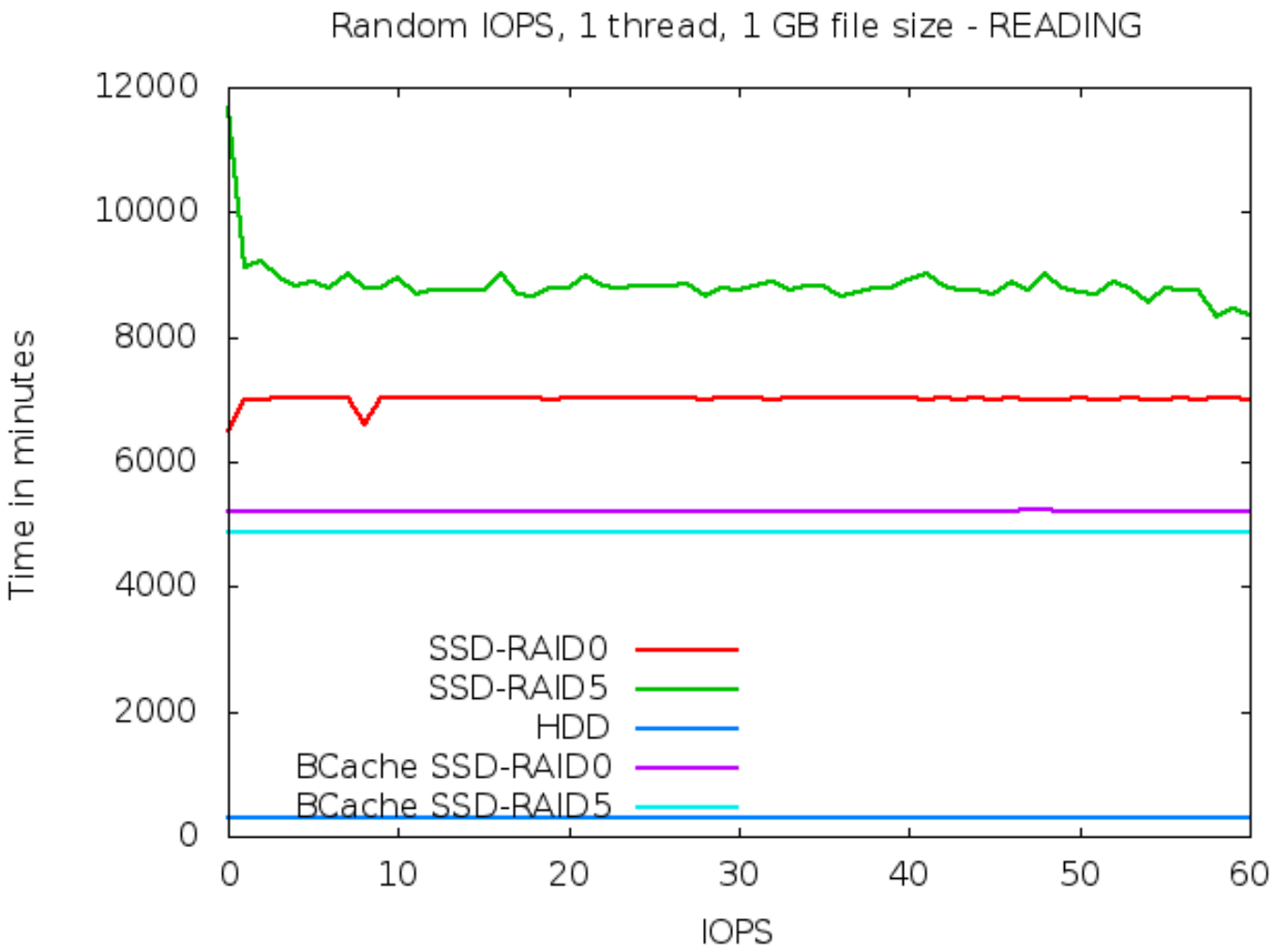

- BCache verliert: sequentiell schreiben von 4k Blöcken ungewöhnlich!

# Vergleich BCache-IOPS seriell schreibend, 10.000 GB



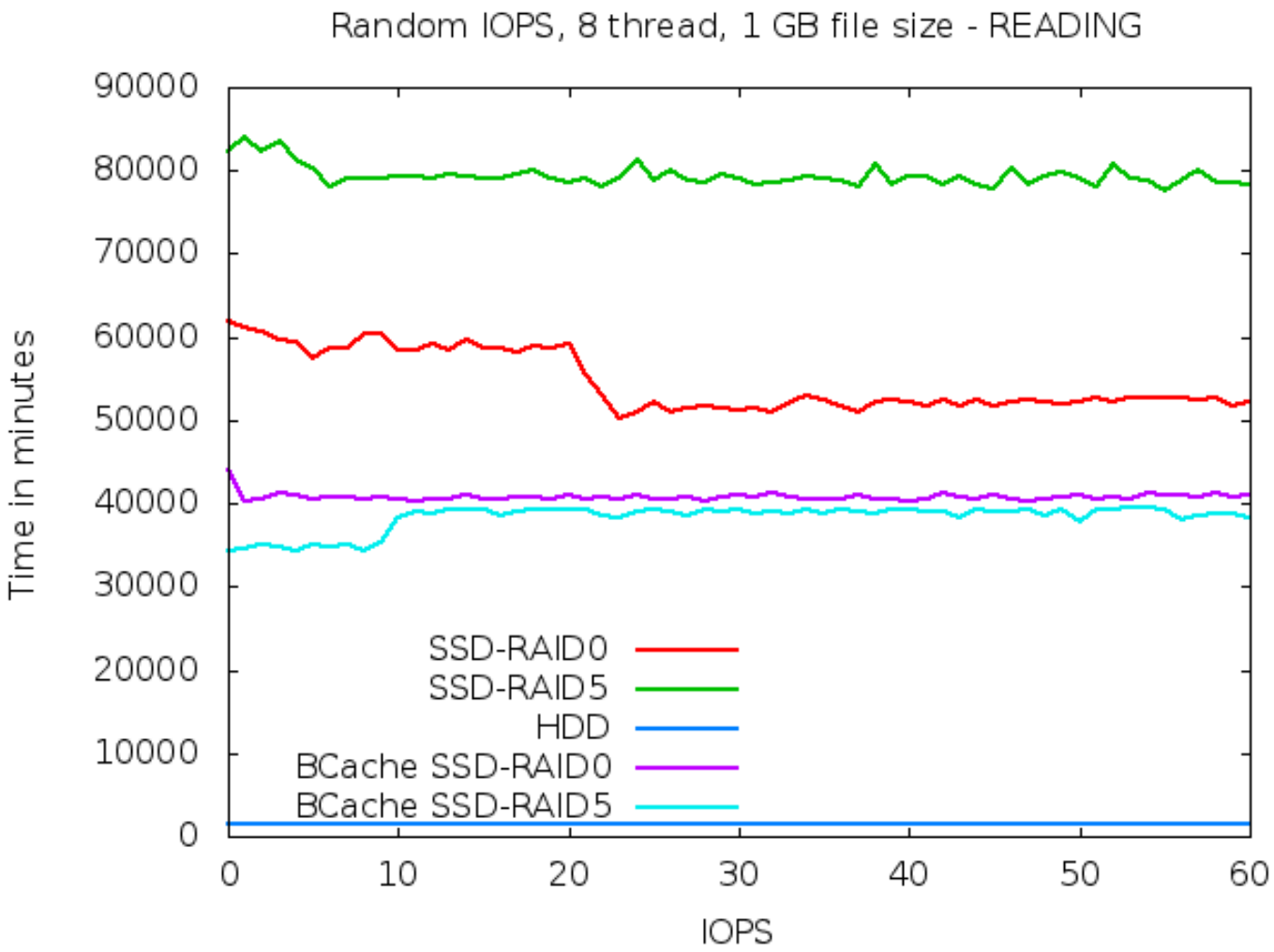

- BCache mit SSD RAID 5 vermutlich wegen doppeltem RAID 5/6
- mehrere Threads: Latenzgewinn der SSDs

# Vergleich BCache-IOPS random lesend, 1 GB



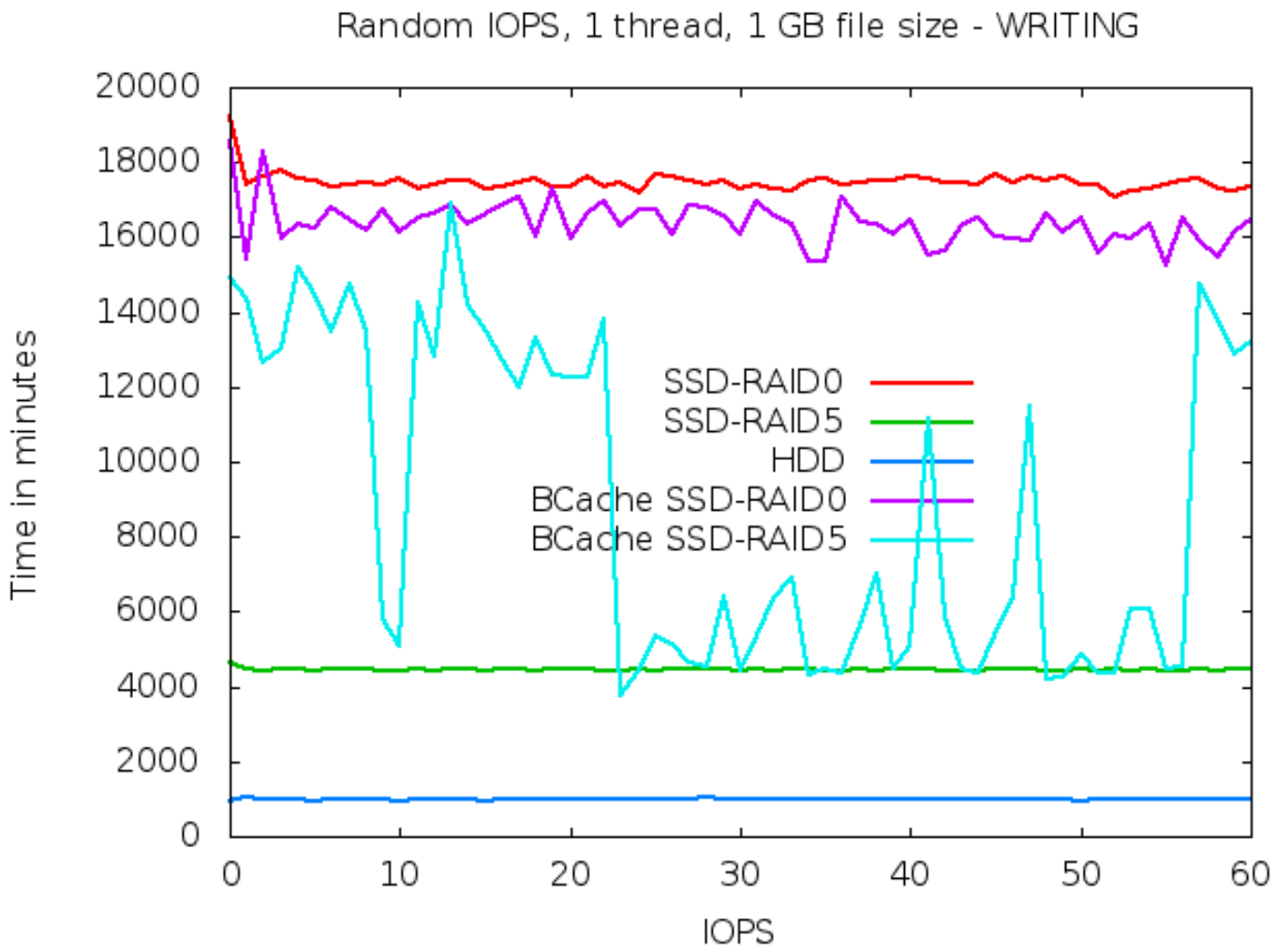

- SSD RAID 5 schneller wegen Striping
- BCache zeigt deutlichen Gewinn

# Vergleich BCache-IOPS random lesend, 1 GB



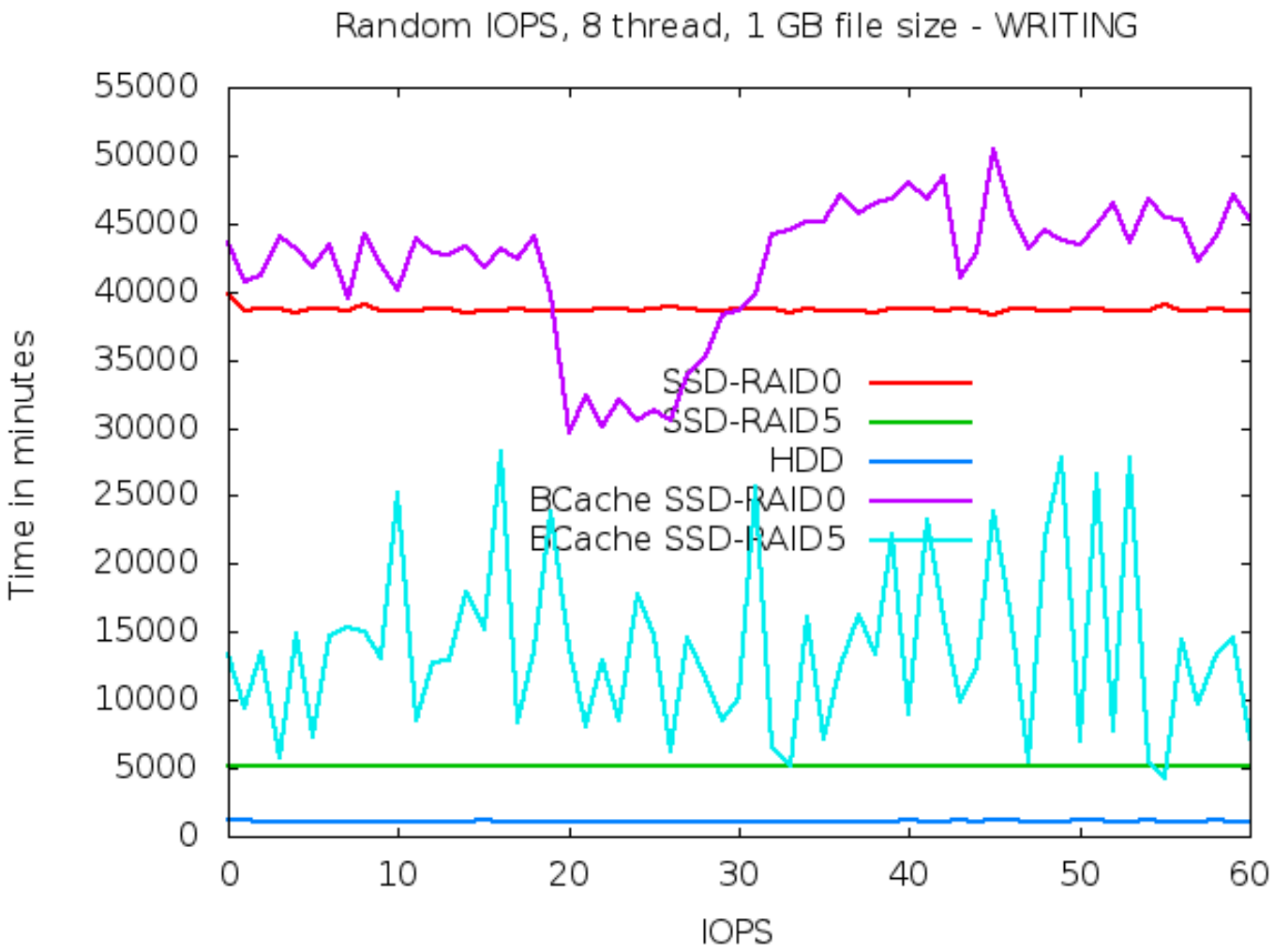

- SSD RAID 5 schneller wegen Striping
- BCache zeigt deutlichen Gewinn

# Vergleich BCache-IOPS random schreibend, 1 GB



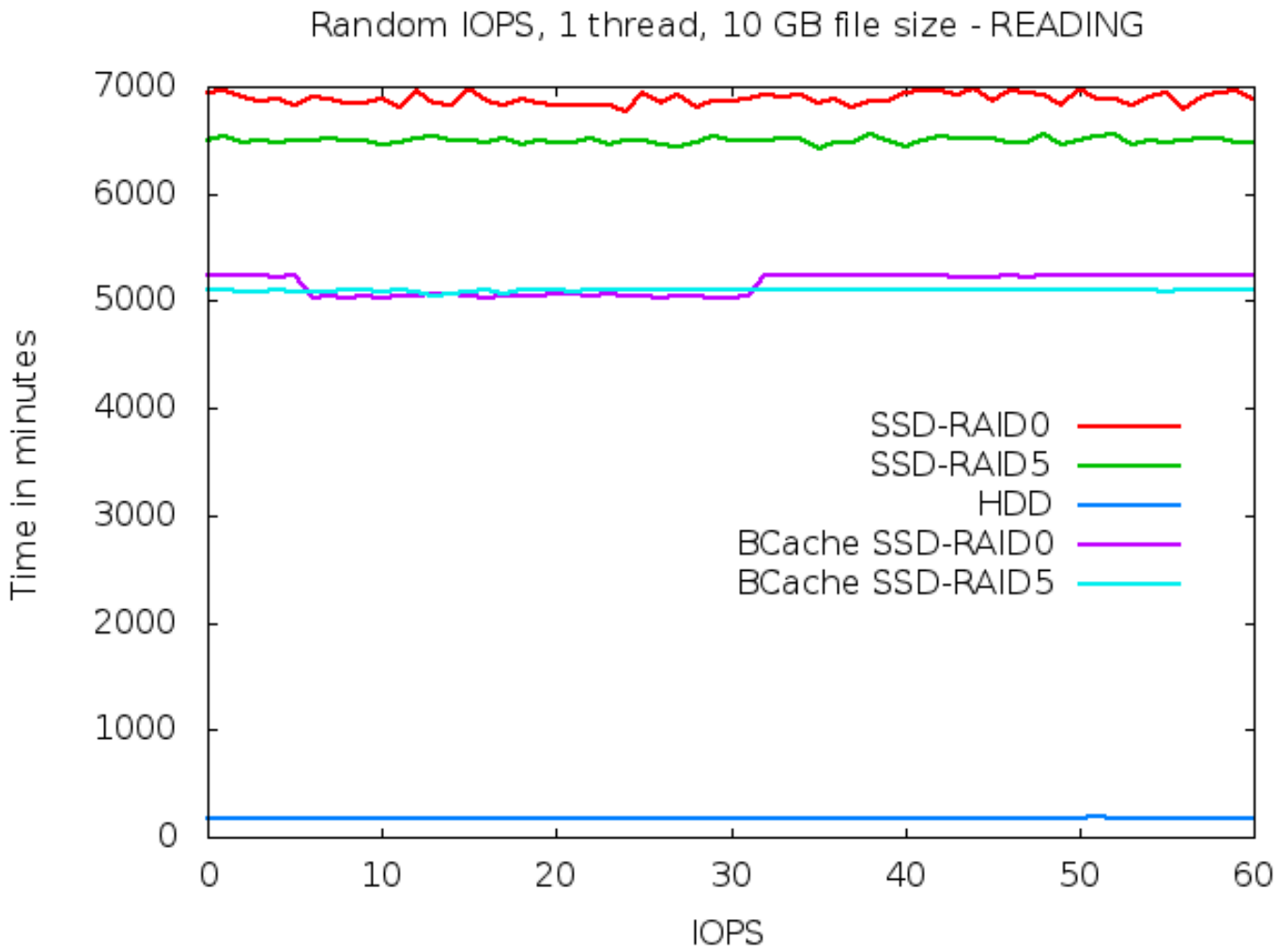

- SSD RAID 0 nun schneller, keine Parity!
- BCache zeigt deutlichen Gewinn!

# Vergleich BCache-IOPS random schreibend, 1 GB



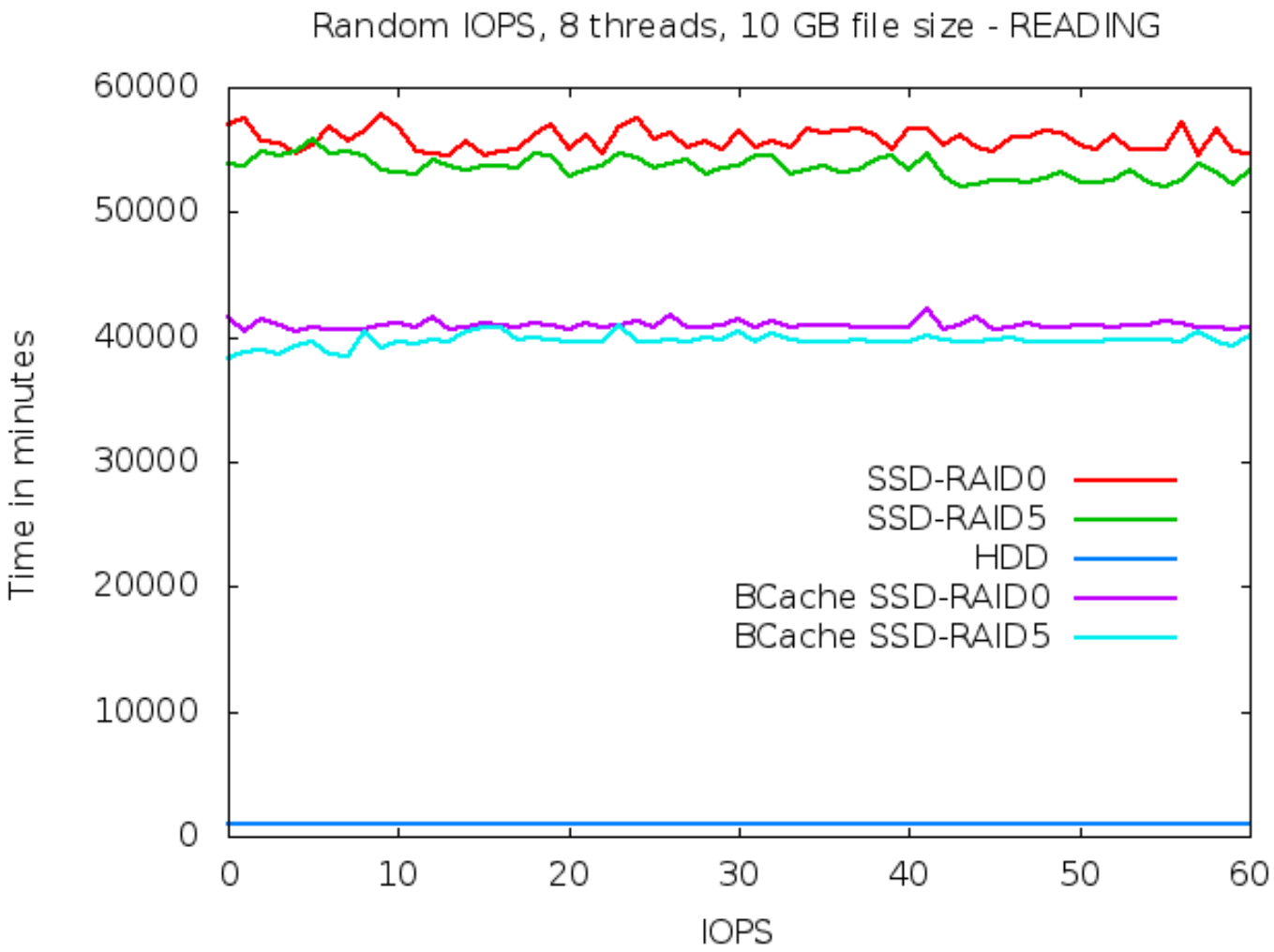

- SSD RAID 0 nun schneller, keine Parity!
- BCache zeigt deutlichen Gewinn, freischaufeln auf HDD

# Vergleich BCache-IOPS random lesend, 10 GB



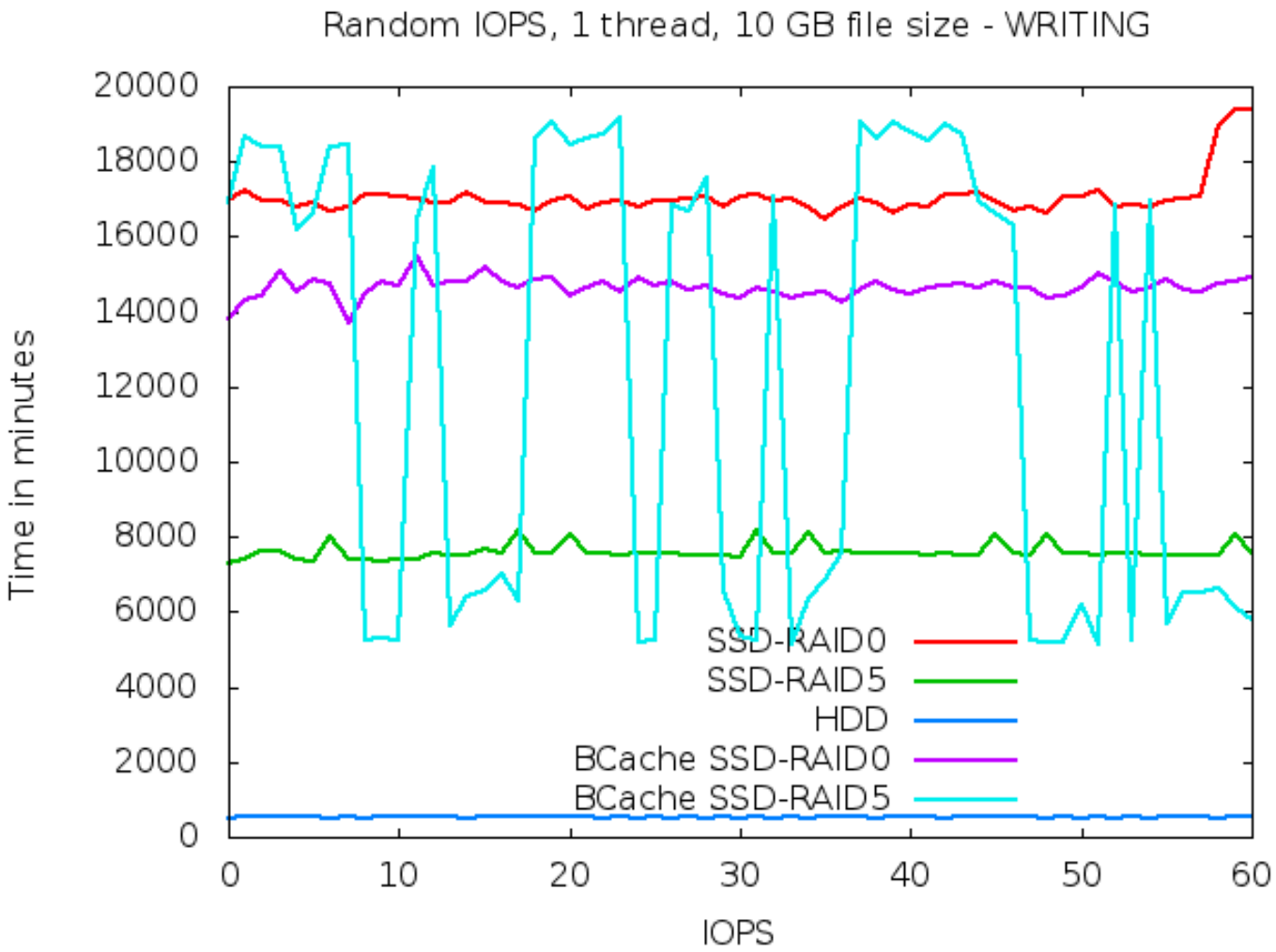

- BCache schneller wegen cache-hits wegen kleiner Datei

# Vergleich BCache-IOPS random lesend, 10 GB



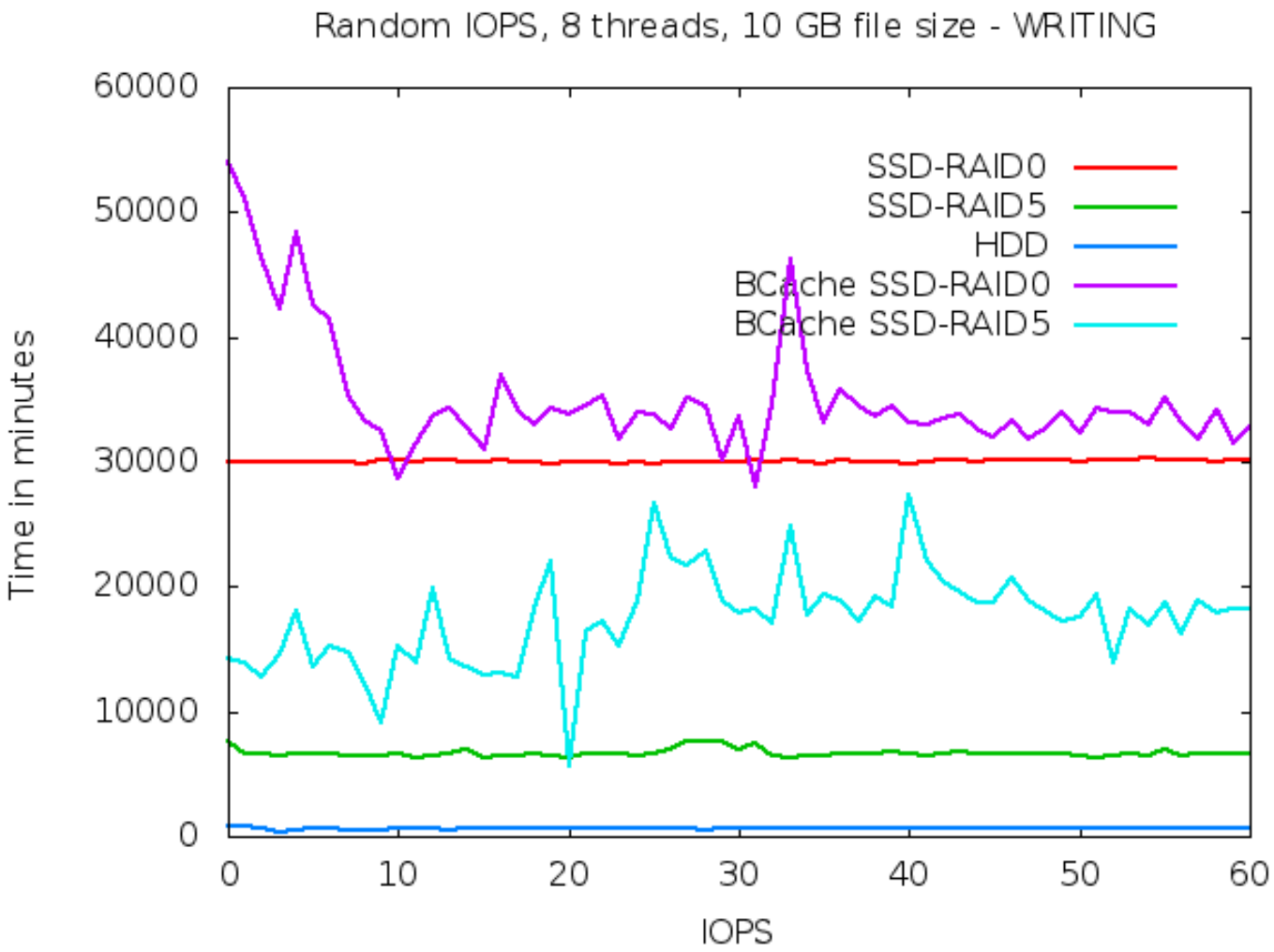

- mehrere Threads: SSDs zeigen Latenzgewinn
- BCache schneller wegen cache-hits wegen kleiner Datei

# Vergleich BCache-IOPS random schreibend, 10 GB



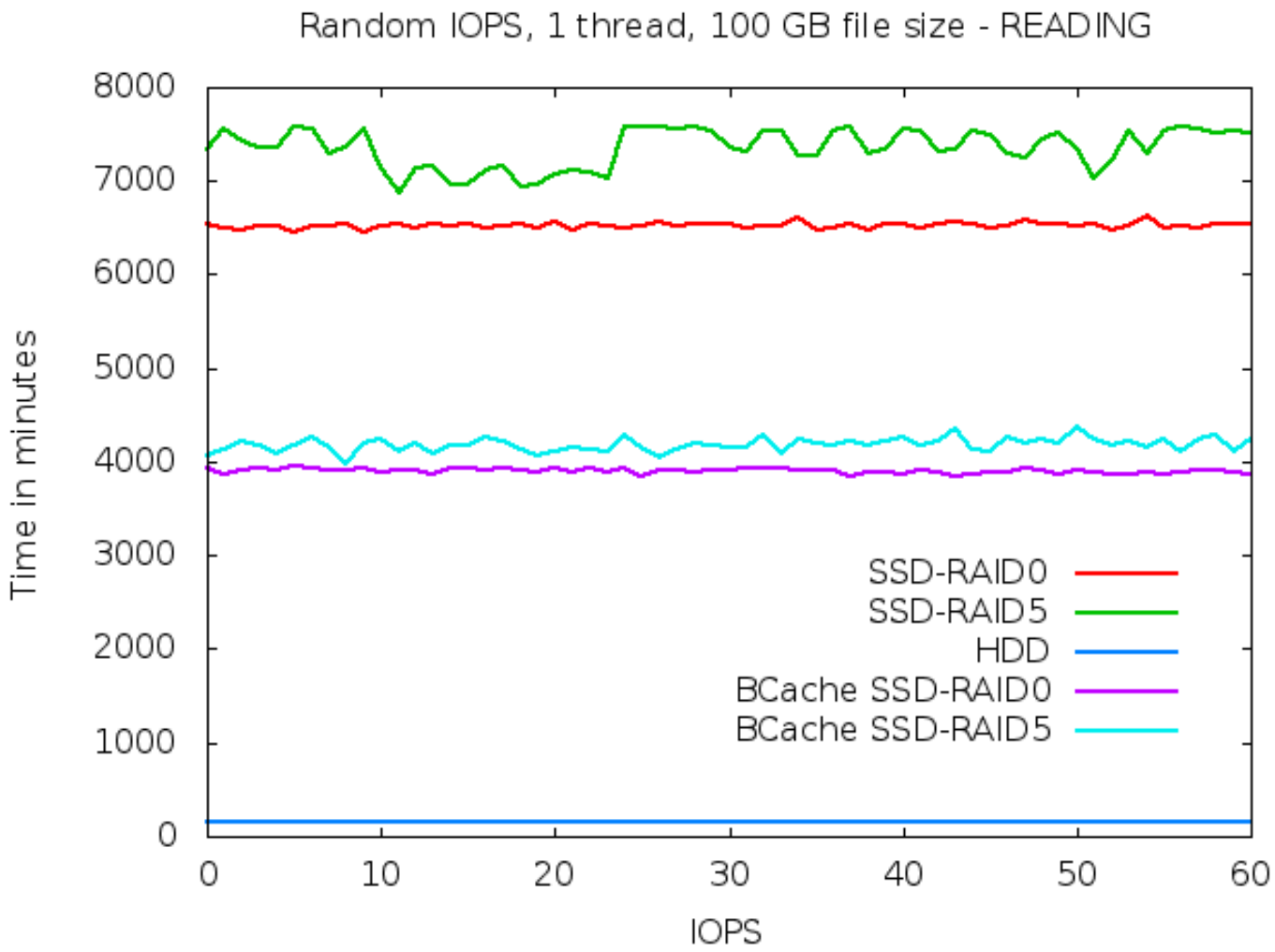

- BCache mit SSD RAID 5 seltsam, vermutlich Parity-Berechnung

# Vergleich BCache-IOPS random schreibend, 10 GB



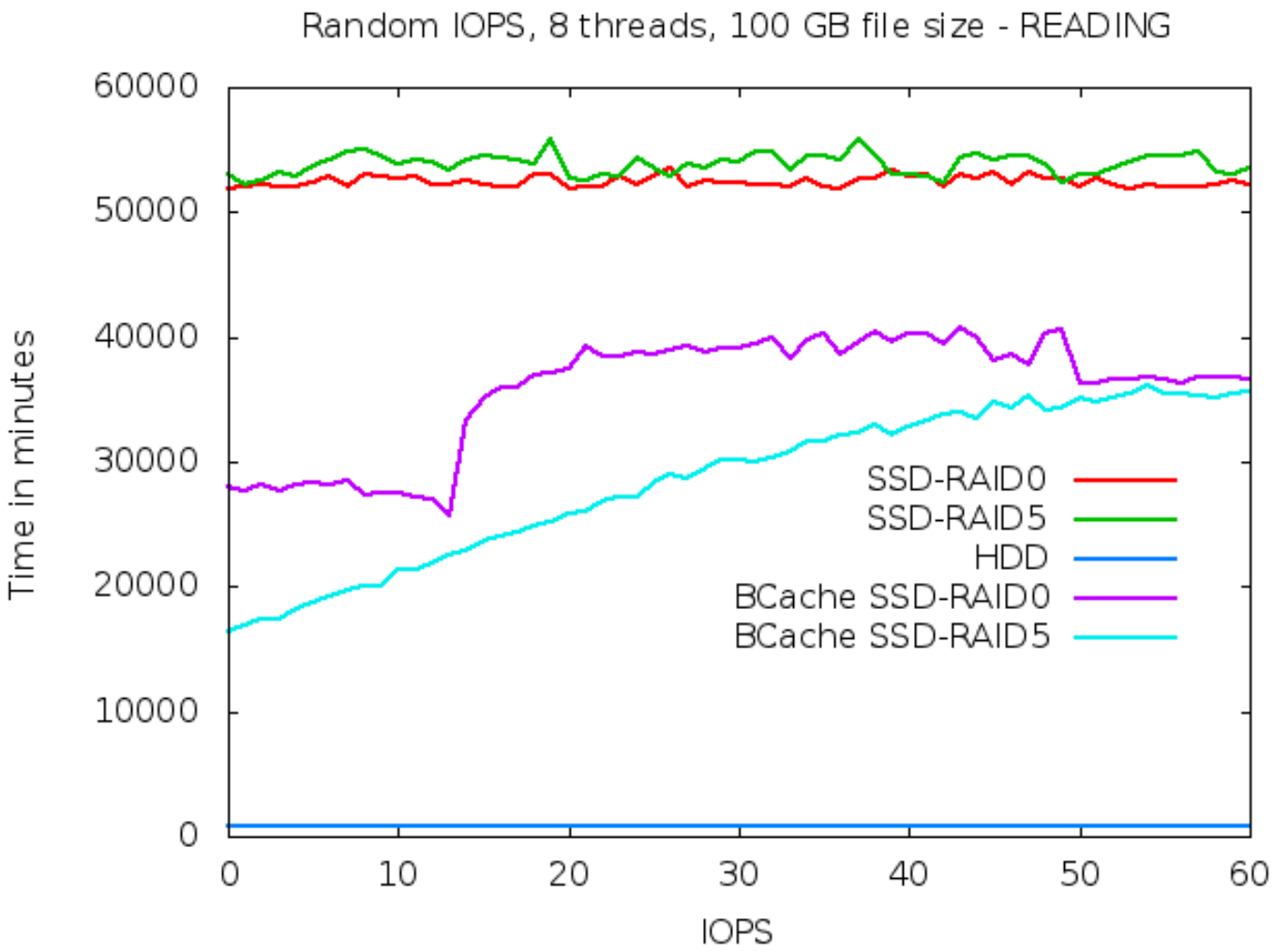

- BCache mit SSD RAID 5 vermutlich wegen doppeltem RAID 5/6
- Übertragen an HDD befreit SSD-Speicher

# Vergleich BCache-IOPS random lesend, 100 GB



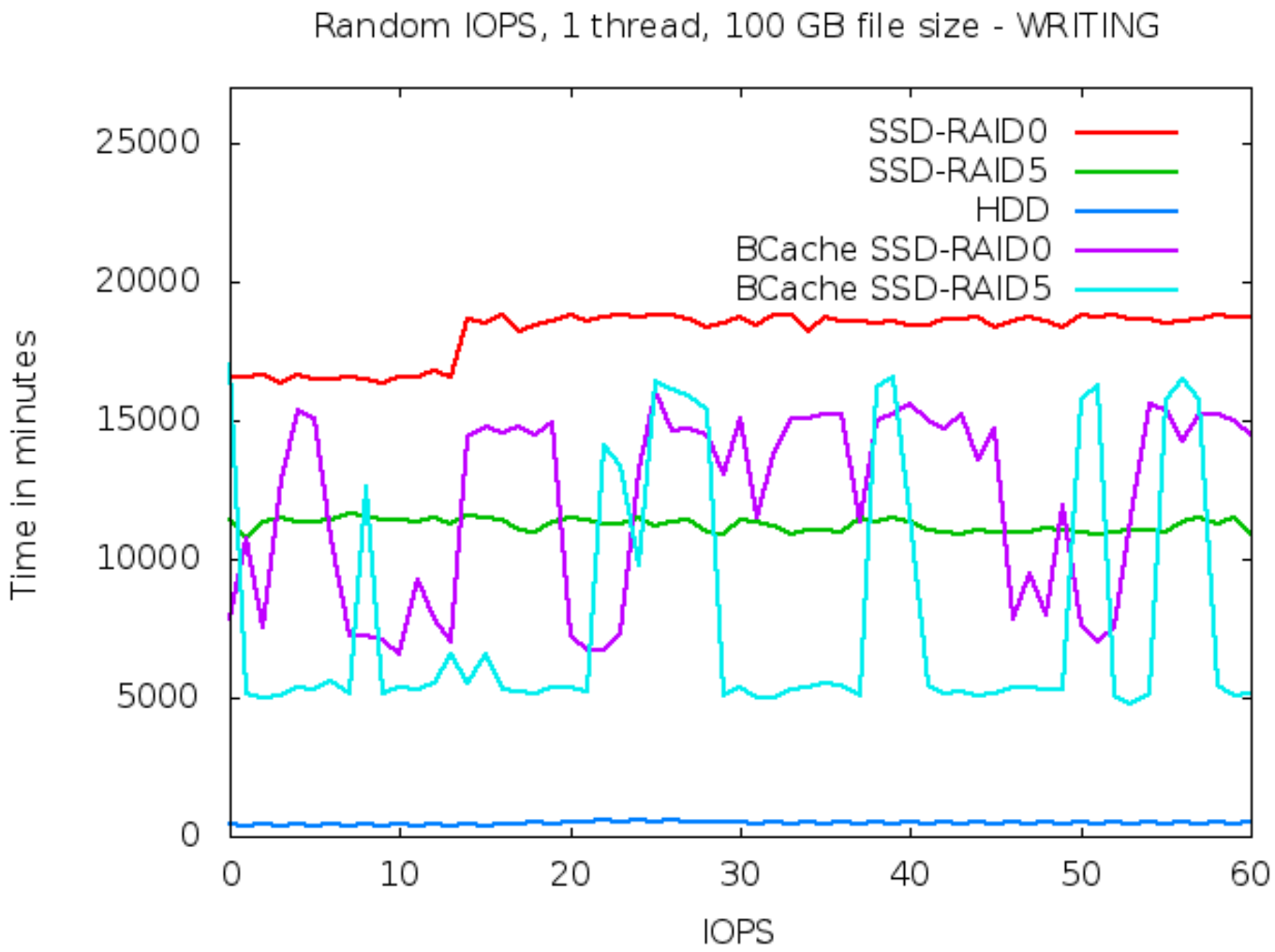

- SSD RAID 5 schneller wegen Striping
- Dateigröße zeigt Wirkung

# Vergleich BCache-IOPS random lesend, 100 GB




- mehrere Threads: Latenzgewinn der SSDs
- Dateigröße zeigt Wirkung

# Vergleich BCache-IOPS random schreibend, 100 GB

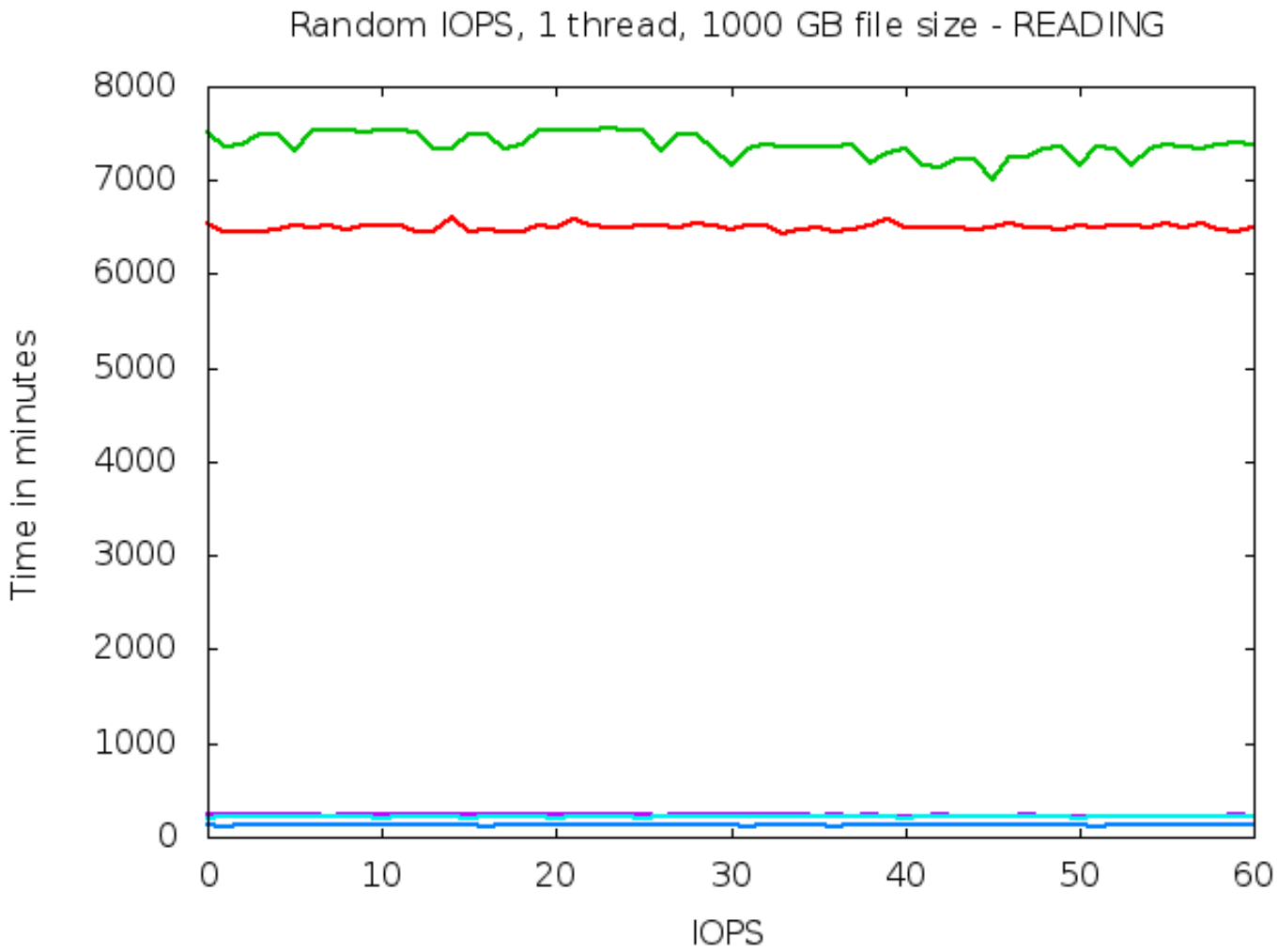



- BCache mit SSD RAID 5 seltsam, vermutlich Parity-Berechnung
- BCache zeigt seine Stärken

# Vergleich BCache-IOPS random schreibend, 100 GB

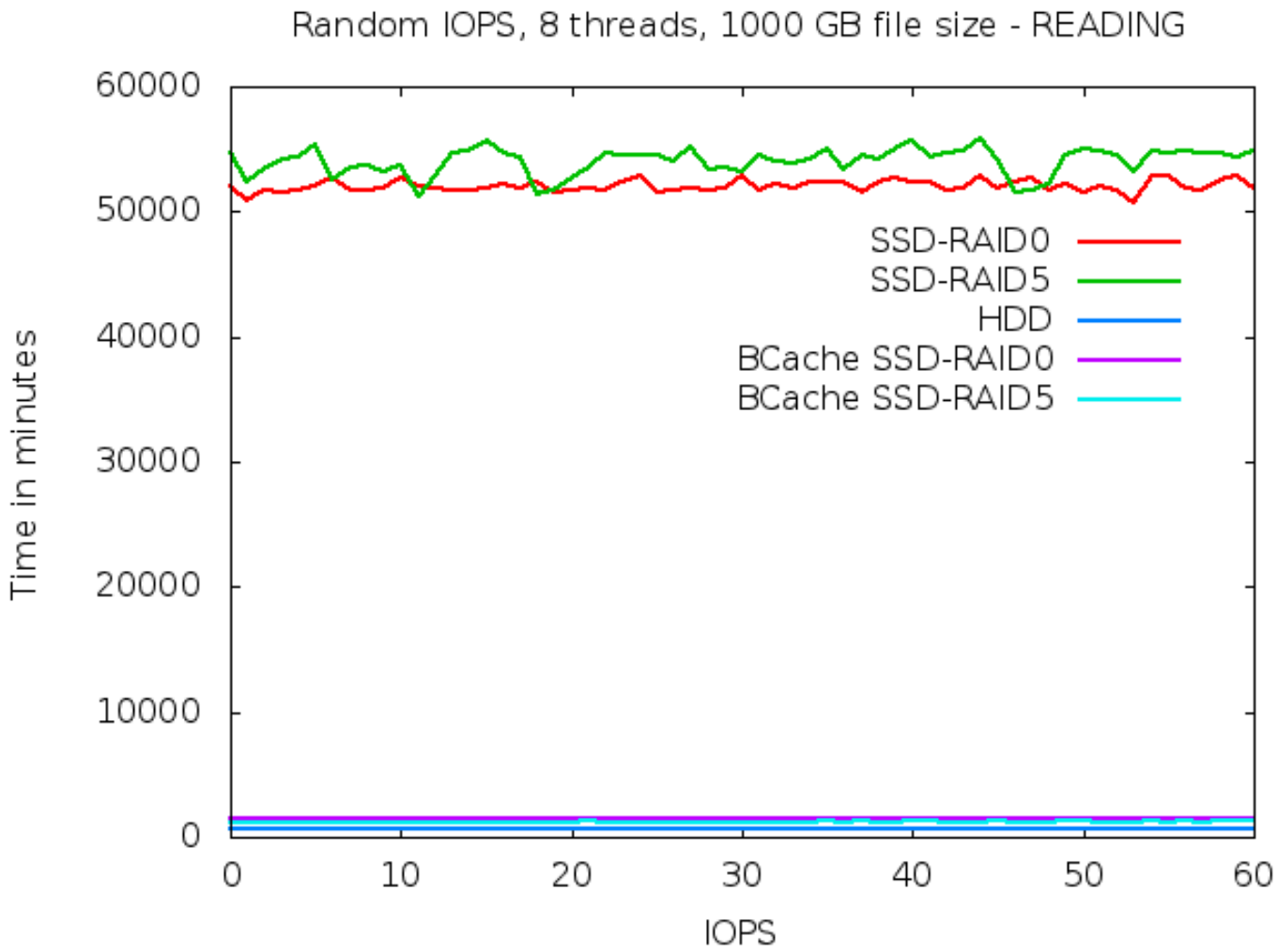

- BCache mit SSD RAID 5 seltsam, vermutlich Parity-Berechnung
- BCache zeigt seine Stärken

# Vergleich BCache-IOPS random lesend, 1000 GB




- BCache verliert, Datei zu groß um Cache schnell zu füllen

# Vergleich BCache-IOPS random lesend, 1000 GB

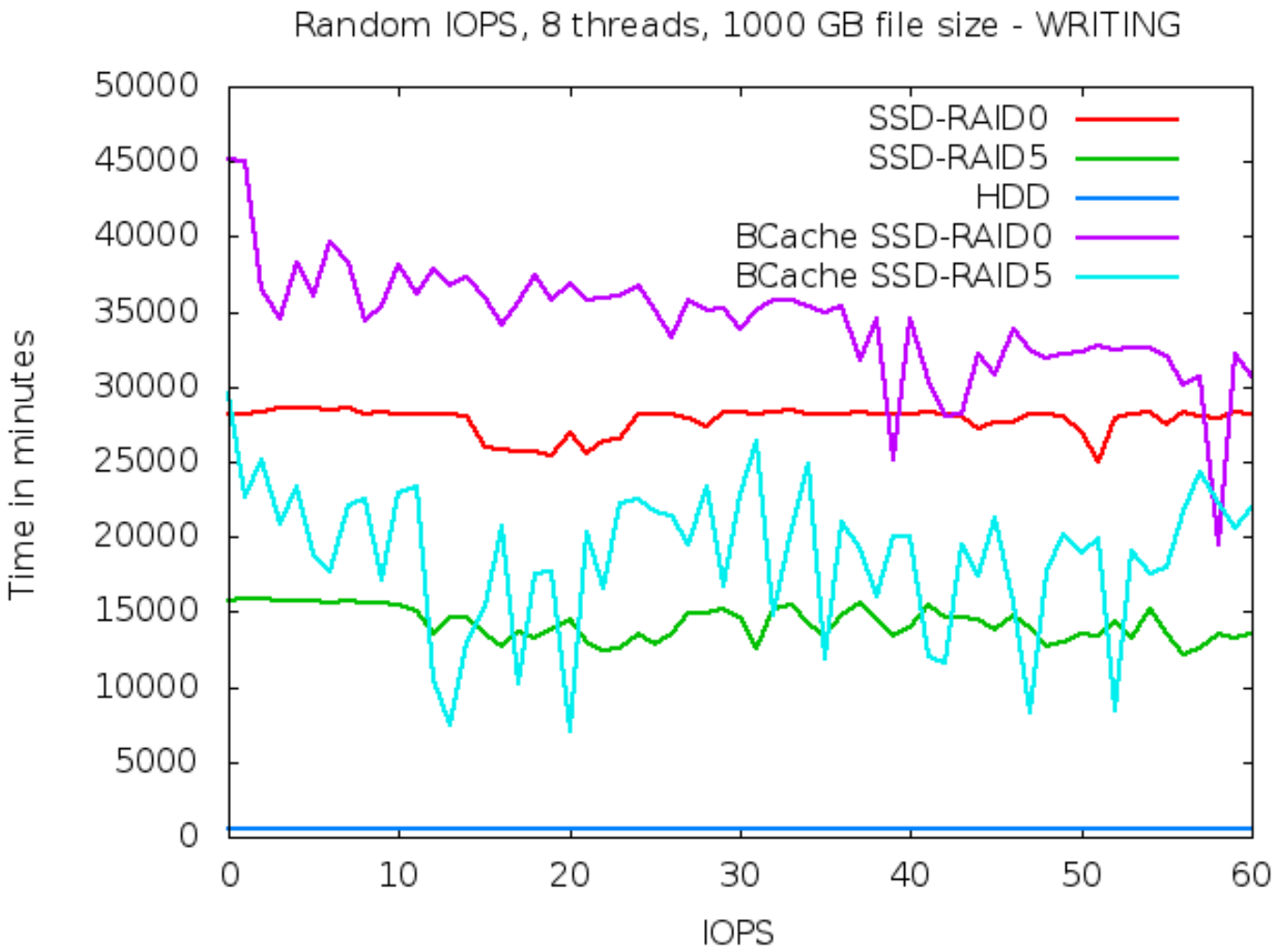



- BCache verliert, Datei zu groß um Cache schnell zu füllen

# Vergleich BCache-IOPS random schreibend, 1000 GB

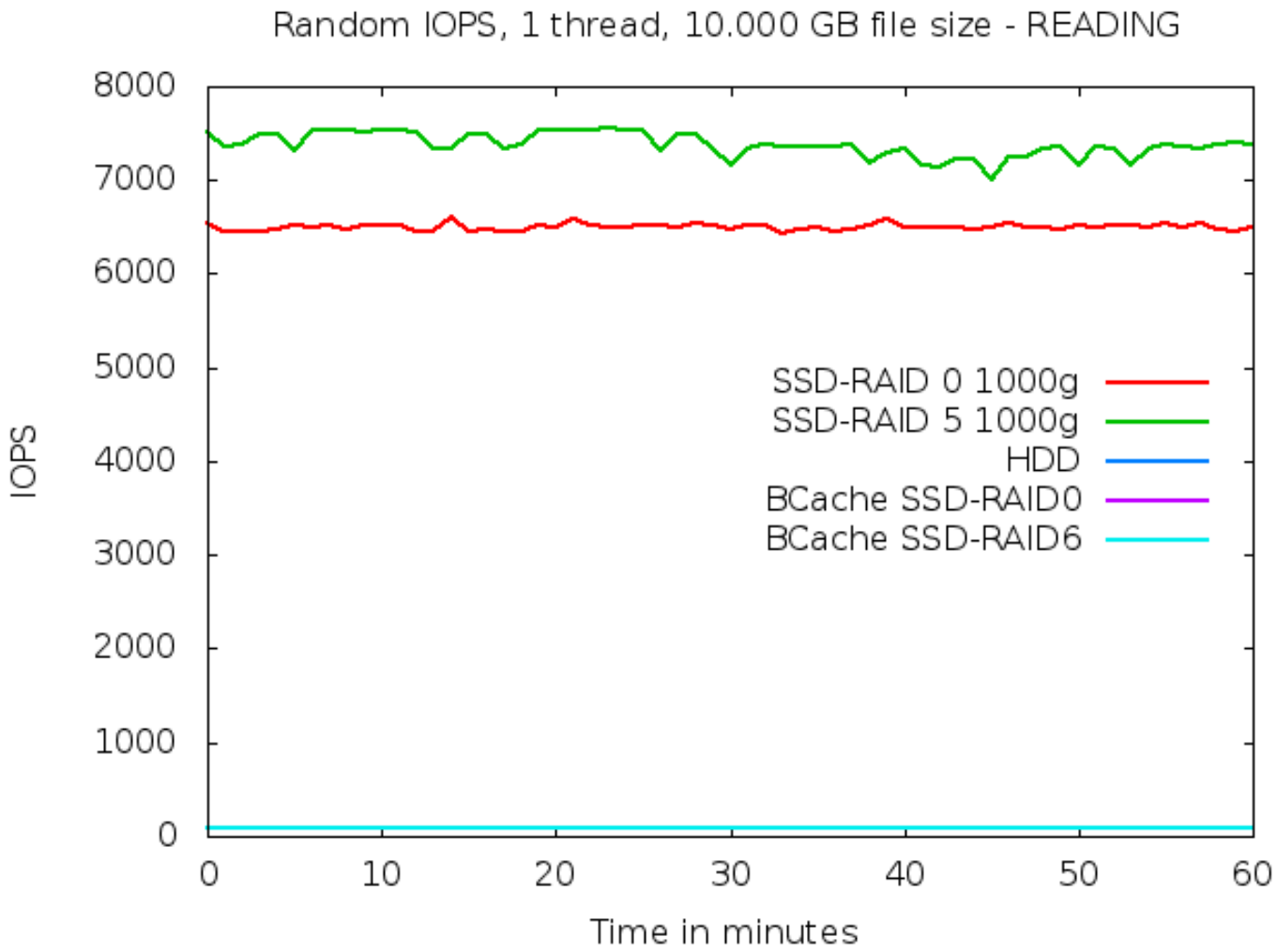

- BCache zeigt seine Stärken!

# Vergleich BCache-IOPS random schreibend, 1000 GB



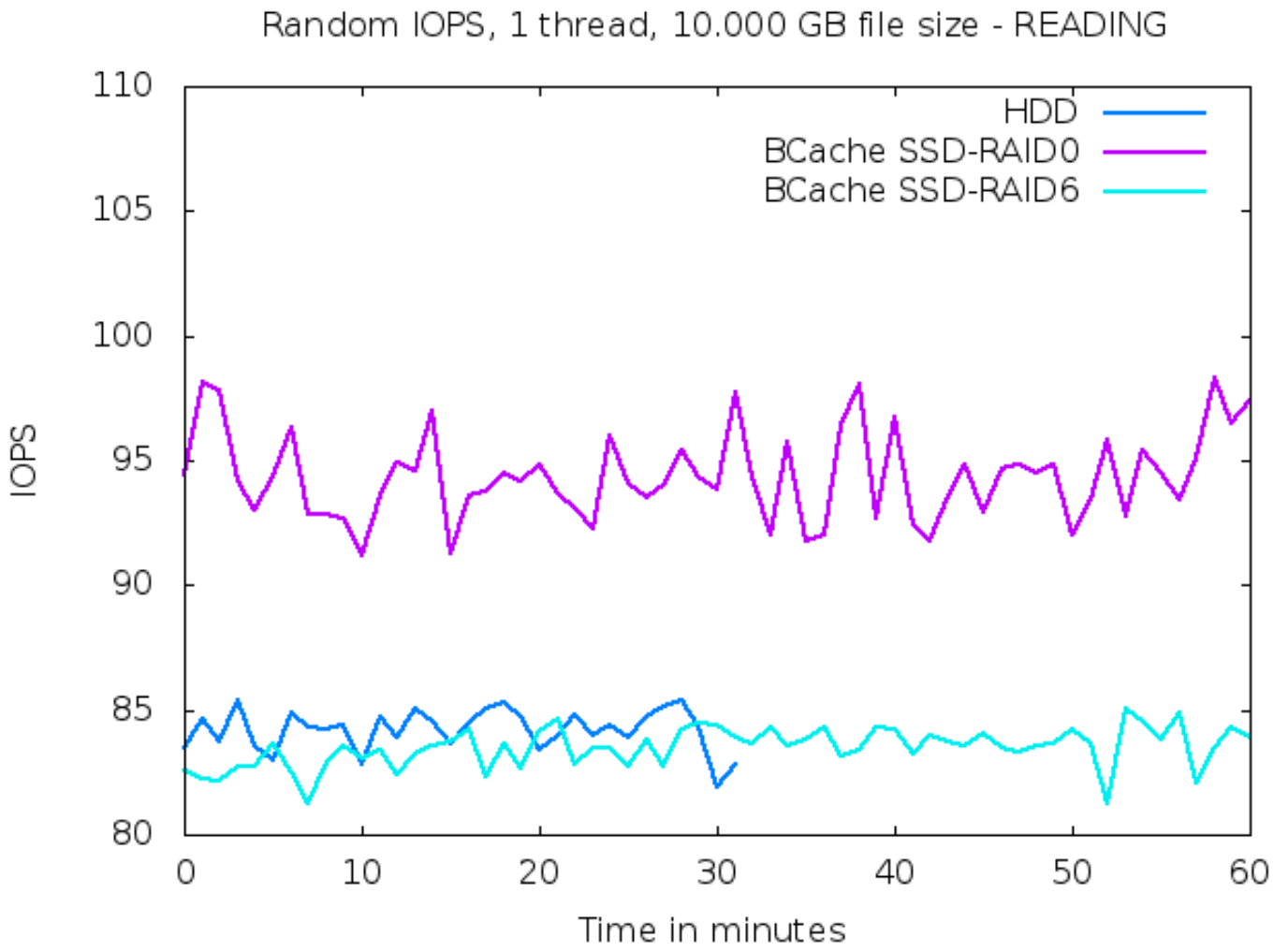

- BCache zeigt seine Stärken!

# Vergleich BCache-IOPS random lesend, 10k GB



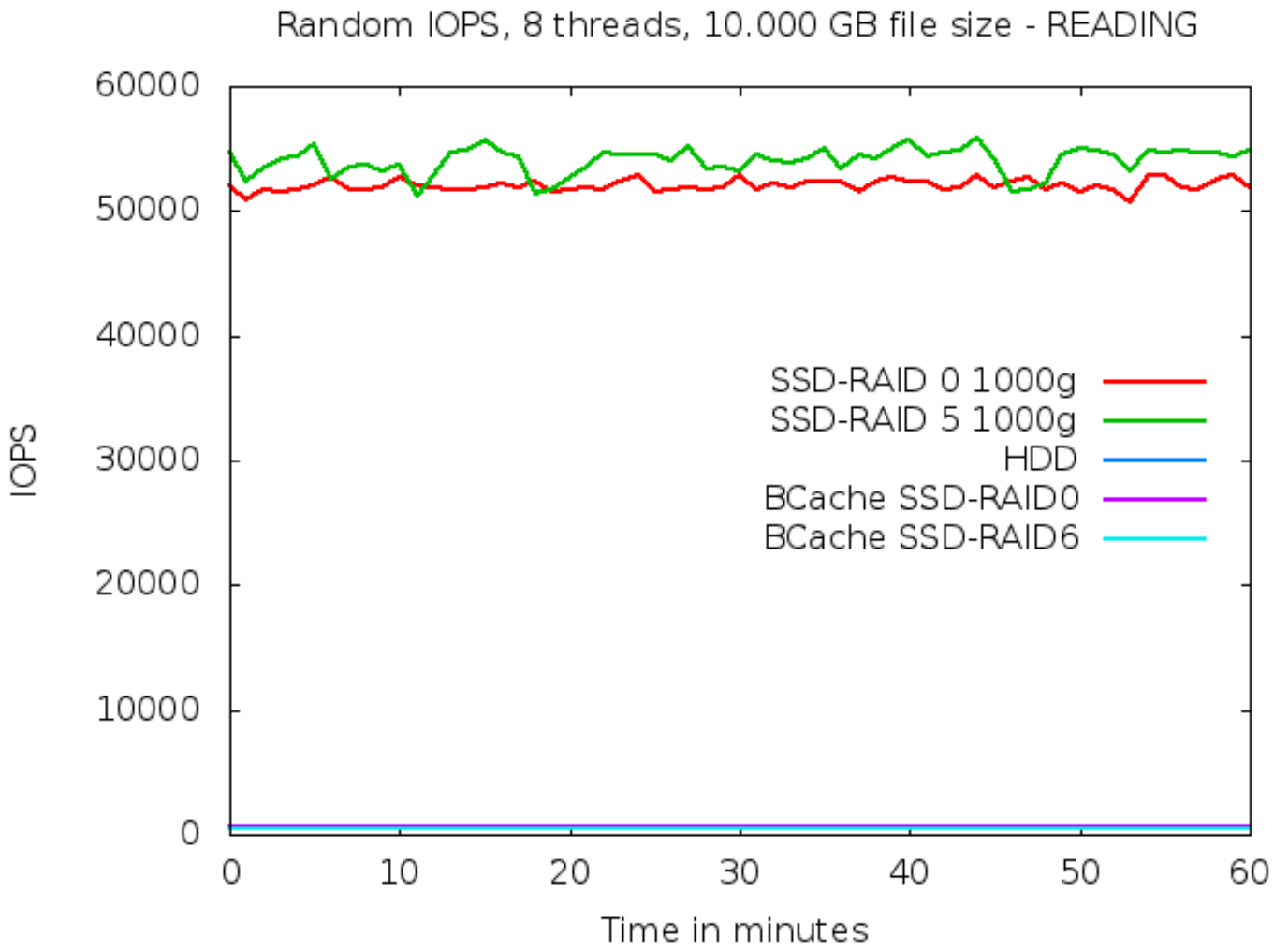

- BCache verliert, Datei zu groß um Cache schnell zu füllen

# Vergleich BCache-IOPS random lesend, 10k GB



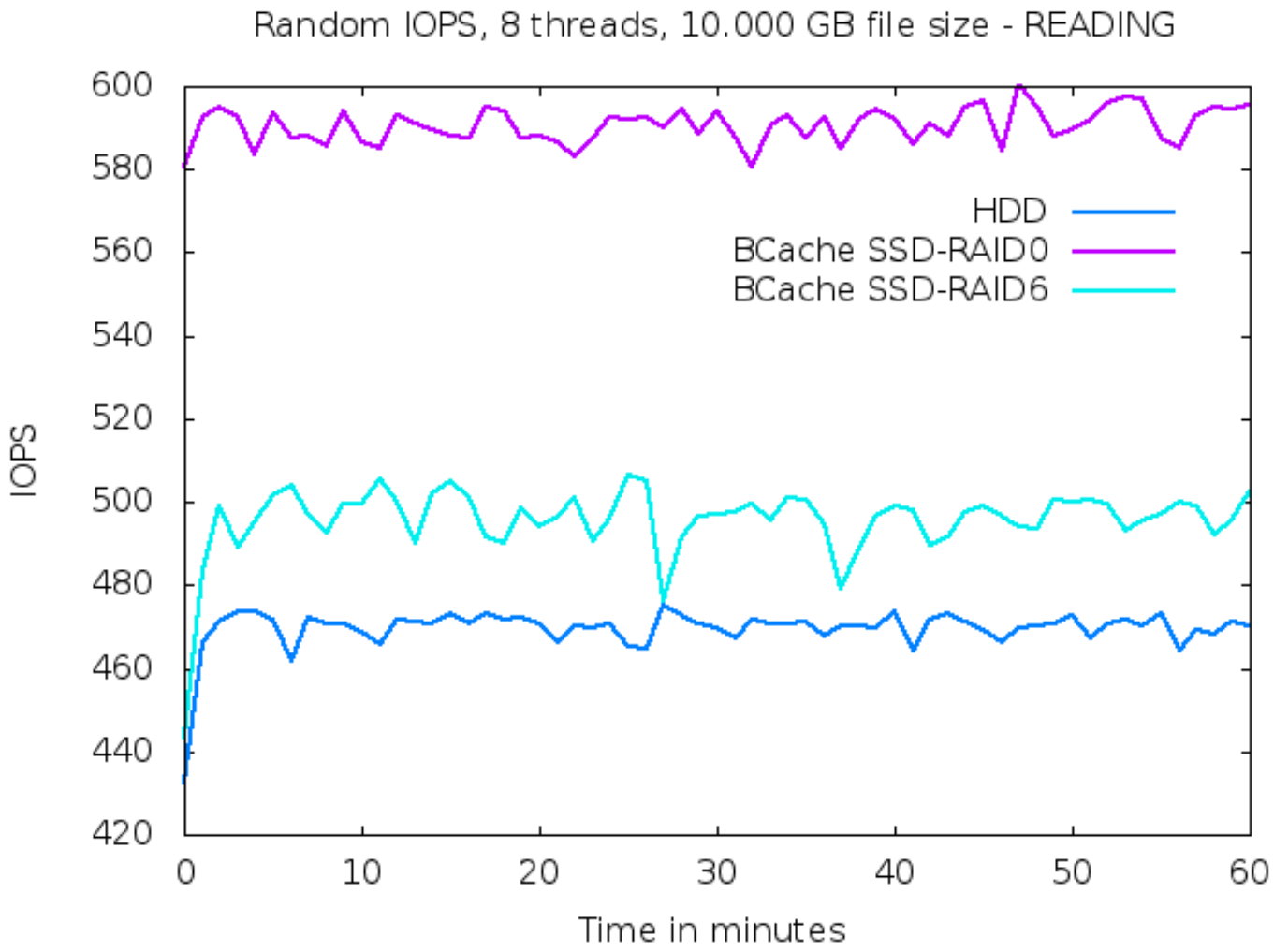

- BCache verliert, Datei zu groß um Cache schnell zu füllen

# Vergleich BCache-IOPS random lesend, 10k GB




- BCache verliert, Datei zu groß um Cache schnell zu füllen

# Vergleich BCache-IOPS random lesend, 10k GB

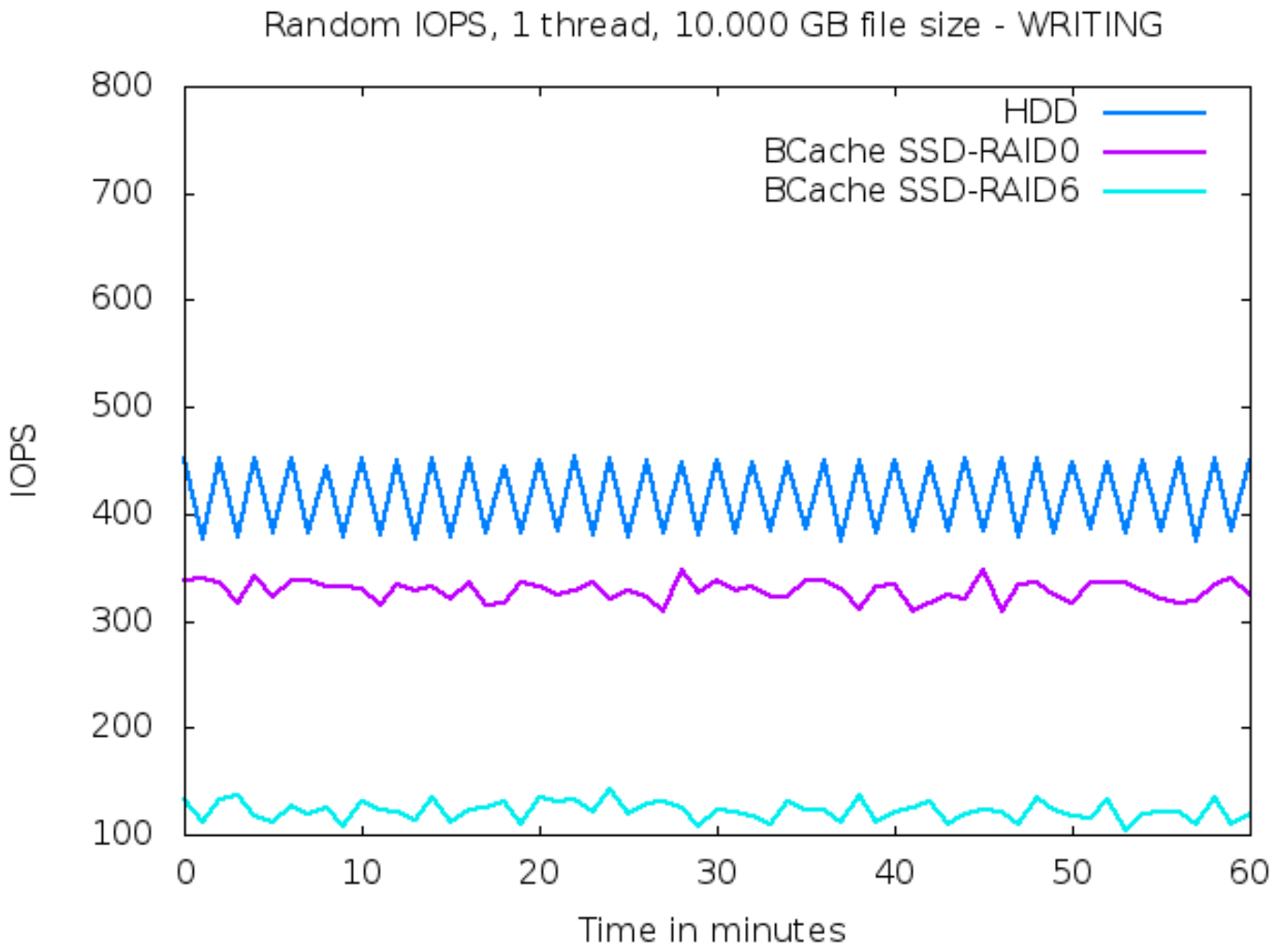



- BCache verliert, Datei zu groß um Cache schnell zu füllen

# Vergleich BCache-IOPS random schreibend, 10k GB

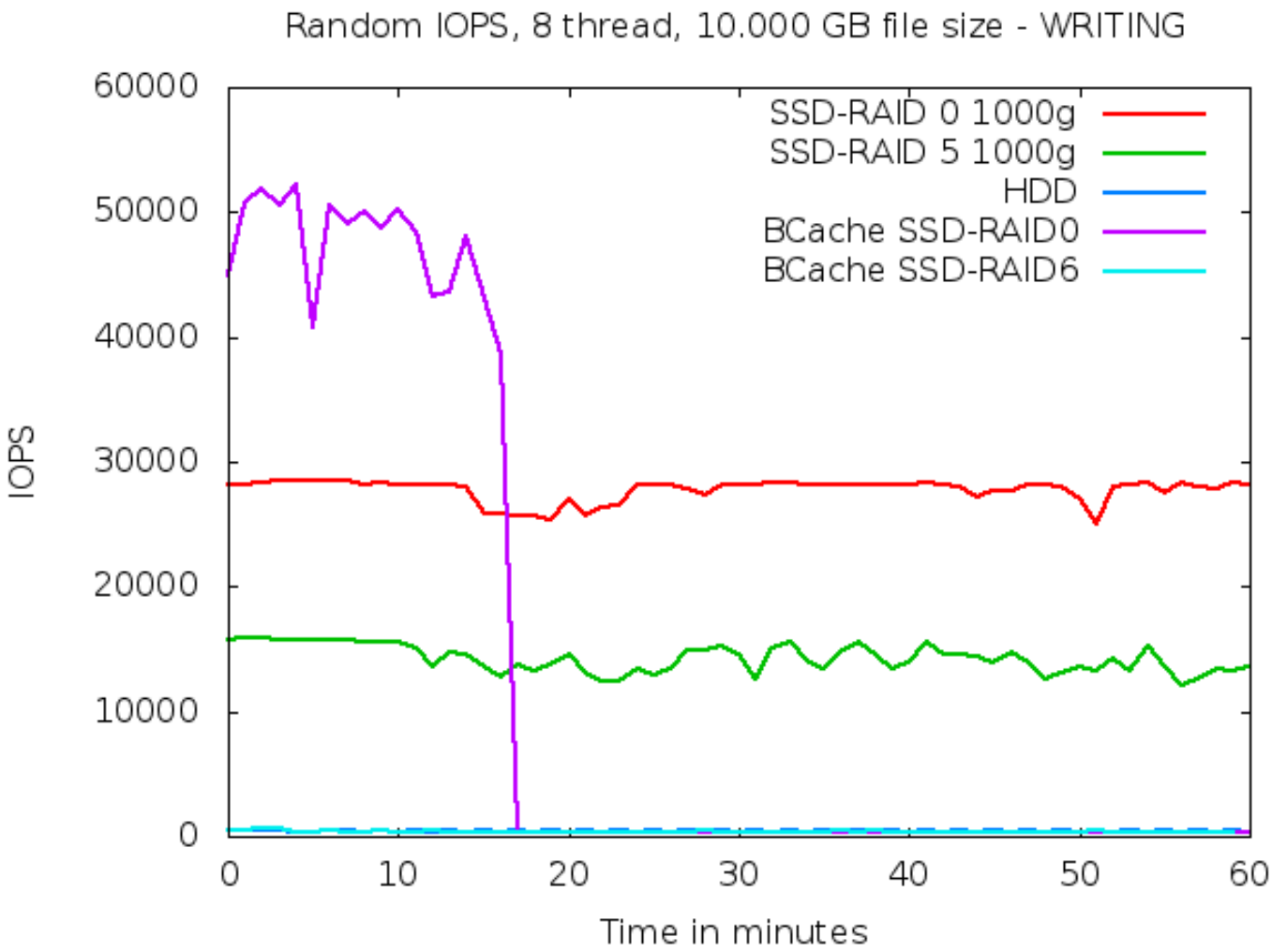

- BCache zeigt keinen Effekt!

# Vergleich BCache-IOPS random schreibend, 10k GB



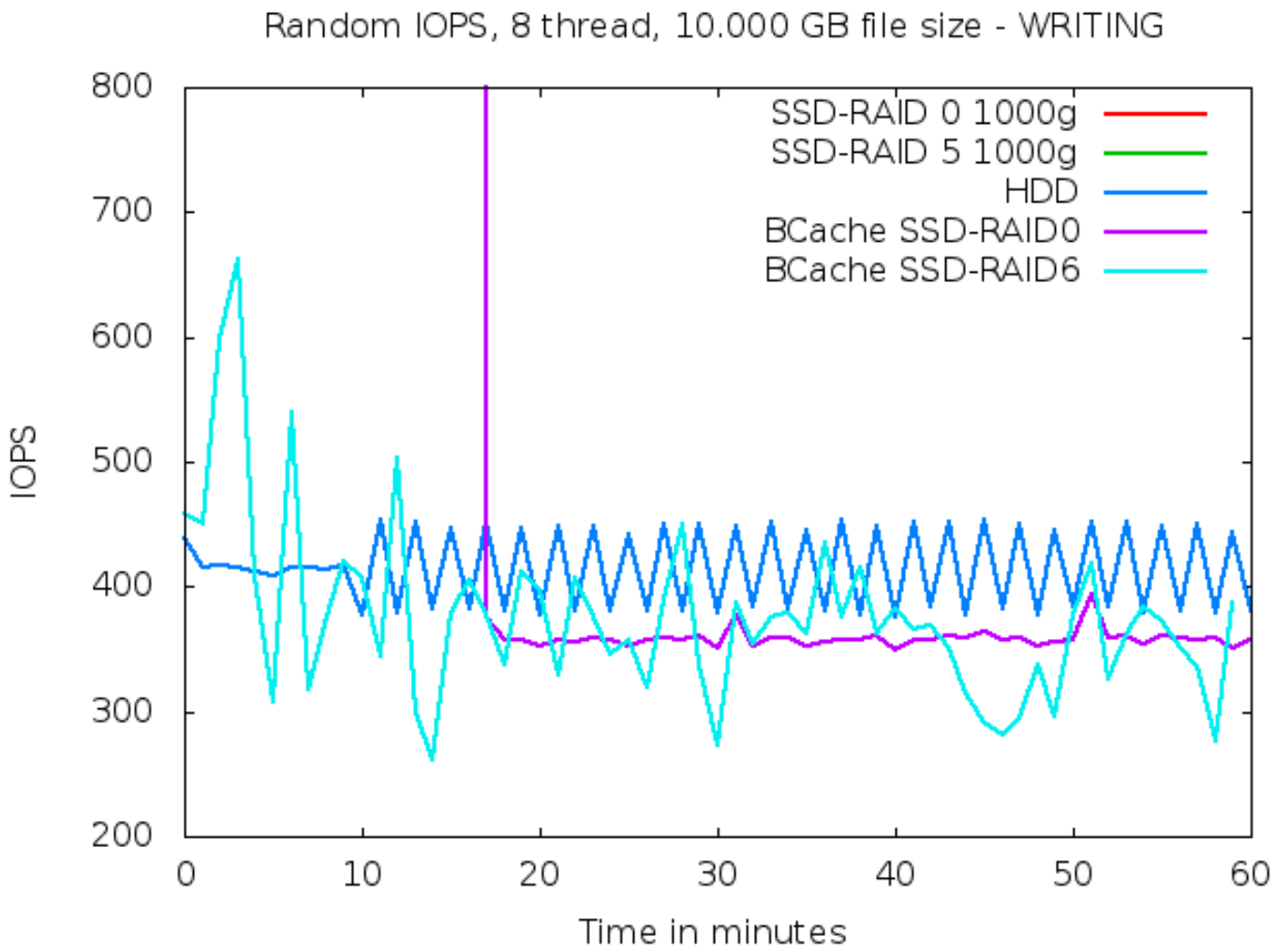

- BCache zeigt keinen Effekt!

# Vergleich BCache-IOPS random schreibend, 10k GB



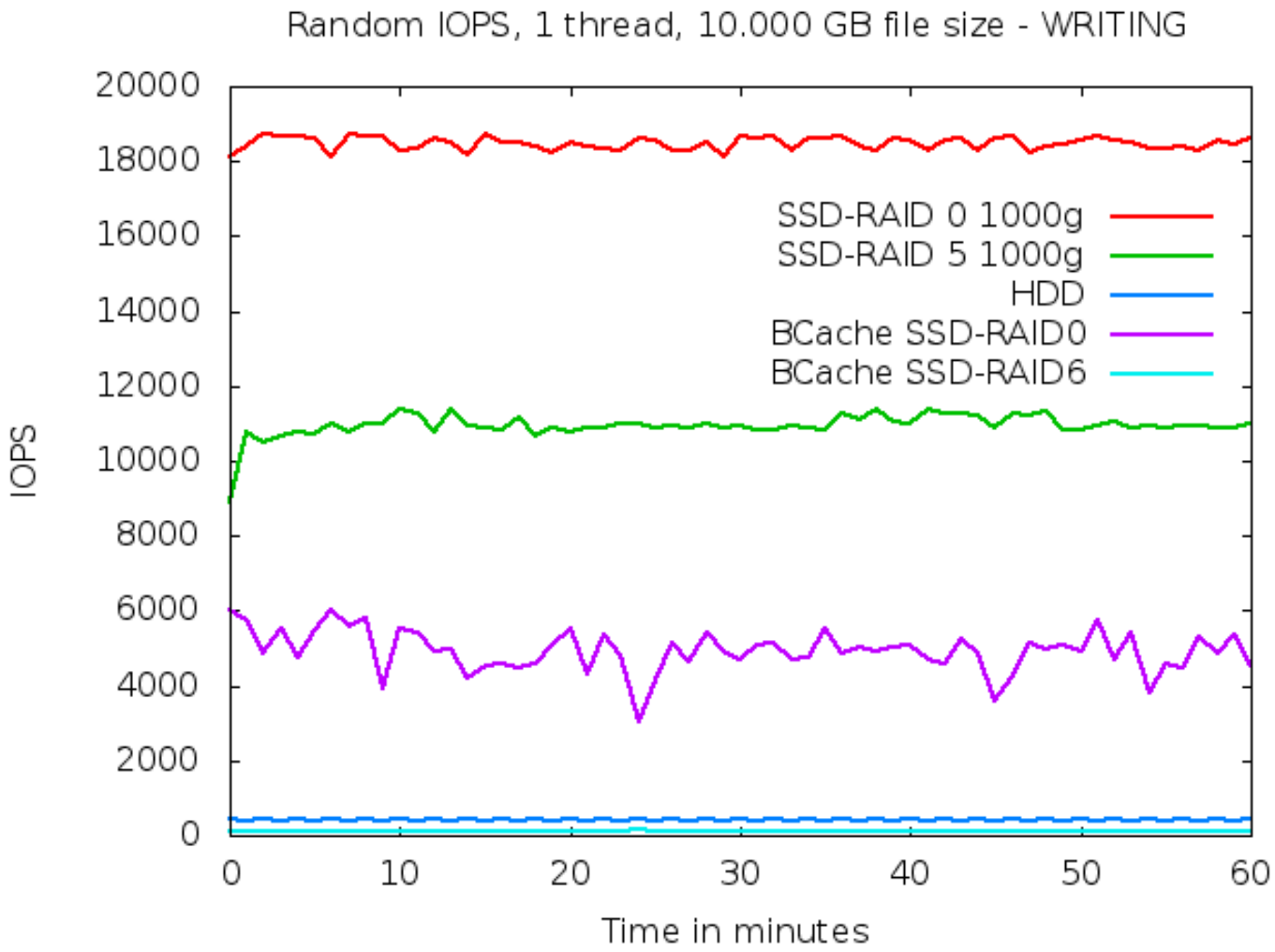

- BCache mit SSD RAID 0 ist am Anfang sehr seltsam. . .
- BCache zeigt ansonsten keinen Effekt!

# Vergleich BCache-IOPS random schreibend, 10k GB



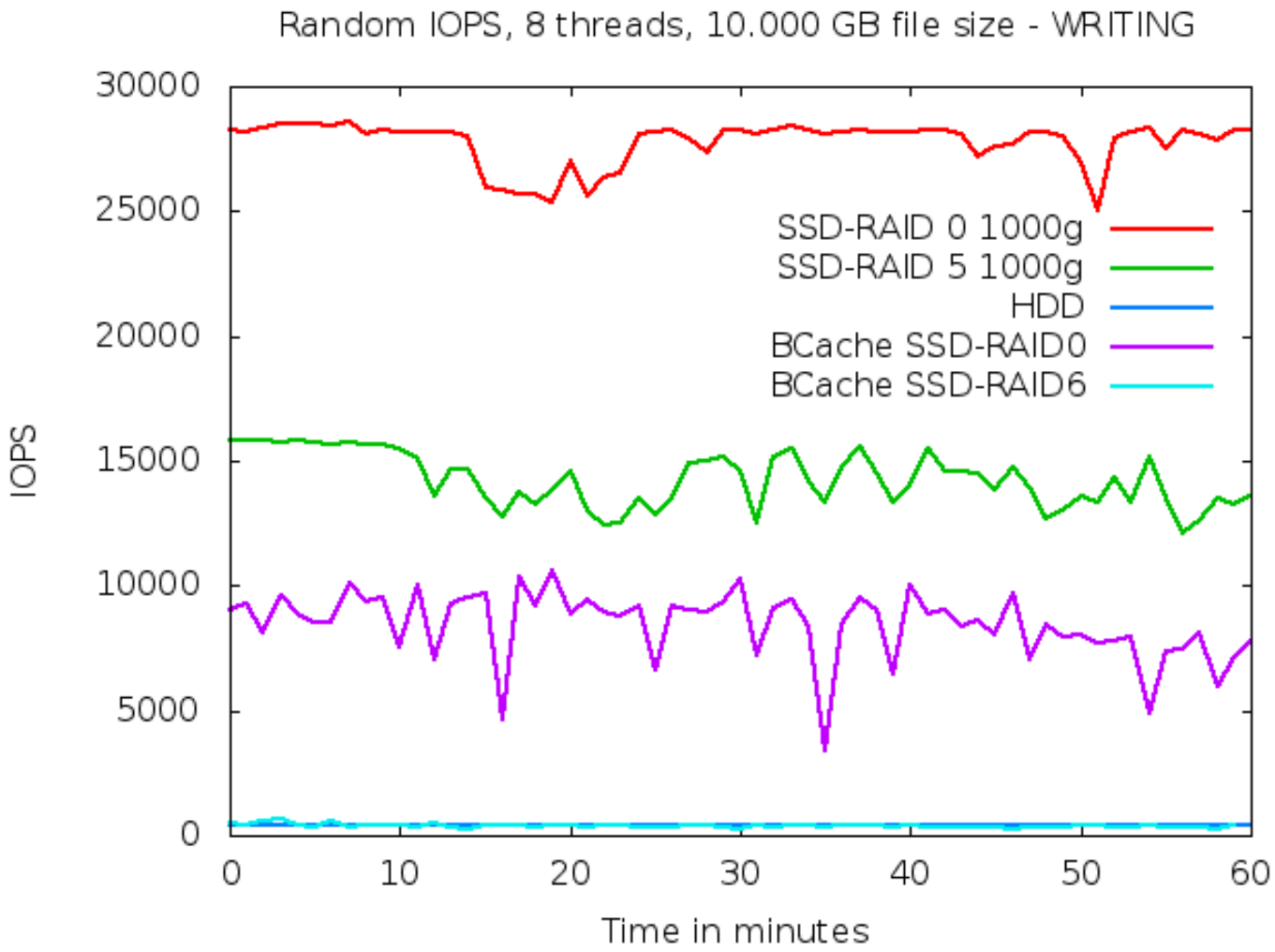

- BCache mit SSD RAID 0 ist am Anfang sehr seltsam. . .
- BCache zeigt ansonsten keinen Effekt!

# Vergleich BCache-IOPS random schreibend, 10k GB




- BCache SSD RAID 0 neu angelegt
- Controller-Problem?

# Vergleich BCache-IOPS random schreibend, 10k GB




- BCache SSD RAID 0 neu angelegt
- Controller-Problem?

# Fazit BCache

- BCache funktioniert eigentlich sehr gut
- unerklärliches Versagen bei sehr großen Dateien, eventuell liegt es am RAID-Controller?
- RAID > 1 für SSDs und HDDs suboptimal
- für WriteBack höhres RAID notwendig
- *work in progress*: BCache ist noch recht neu

# BCache im Notebook

- 1 SSD und 1 HDD
- SSD als BCache im WriteThrough-Modus
- Cached die wichtigsten Dateien zum Booten
- Cached die wichtigsten Dateien im Gebrauch
- sehr schnelles booten möglich und Größe der HDD kann ebenfalls genutzt werden

# Gesamtfazit

- BCache zeigt im Gegensatz zu CacheCade deutliche Vorteile
- CacheCade ist teuer, auf LSI beschränkt und zu klein, wenig wirksam
- BCache-Problem bei großen Dateien könnte auch vom RAID-Controller stammen
- Frage: Ist das ein Anwendungsfall?
  - OS übernimmt ebenfalls Caching/Buffering
  - Hat man viel random I/O mit kleinen Blöcken?
  - bei Datenbanken kann es sehr hilfreich sein: Viele kleine Daten ...
  - Plattenplatz könnte anders genutzt werden
  - bei WriteBack: Redundanz ist wichtig, höherer RAID-Level gegen Ausfall einer SSD
  - Beim Notebook: Ist es das Wert? Man kann die SSD kaum anders nutzen.

# Tja, und nun?