2. E/A-Architekturen und Speicherhierarchien

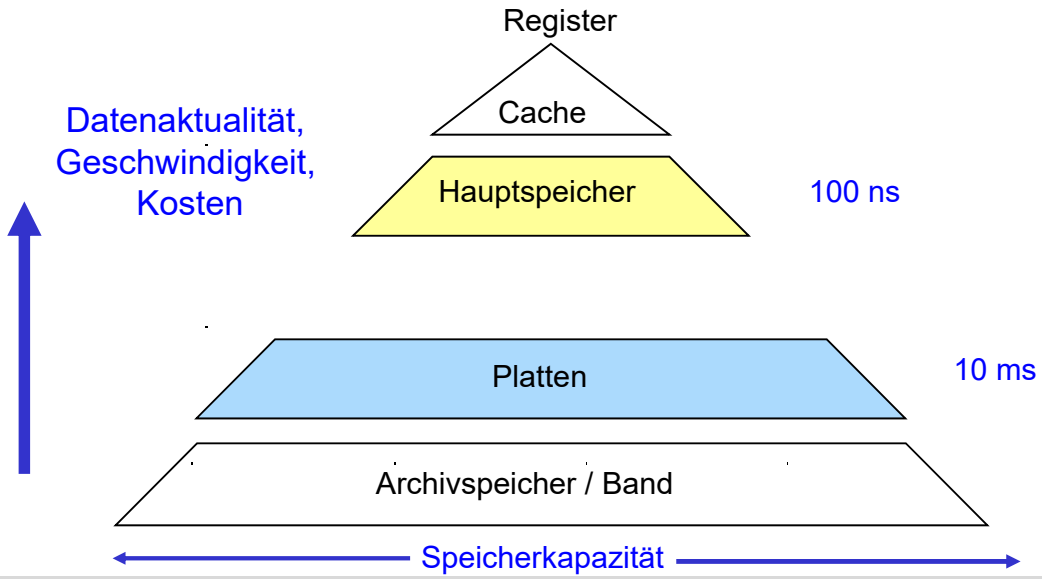
- Einsatz einer Speicherhierarchie
- Magnetplatten
  - technische Merkmale
  - Bestimmung der Zugriffszeit
  - wahlfreie vs. sequenzielle Plattenzugriffe
- Disk-Arrays
  - Realisierungsformen
  - Datenallokation
  - Fehlertoleranz
  - RAID-1 vs. RAID-5
- seitenadressierbare Halbleiterspeicher
  - Platten-Caches
  - Flash-SSD
- 5-Minuten-Regel

Einsatz einer Speicherhierarchie

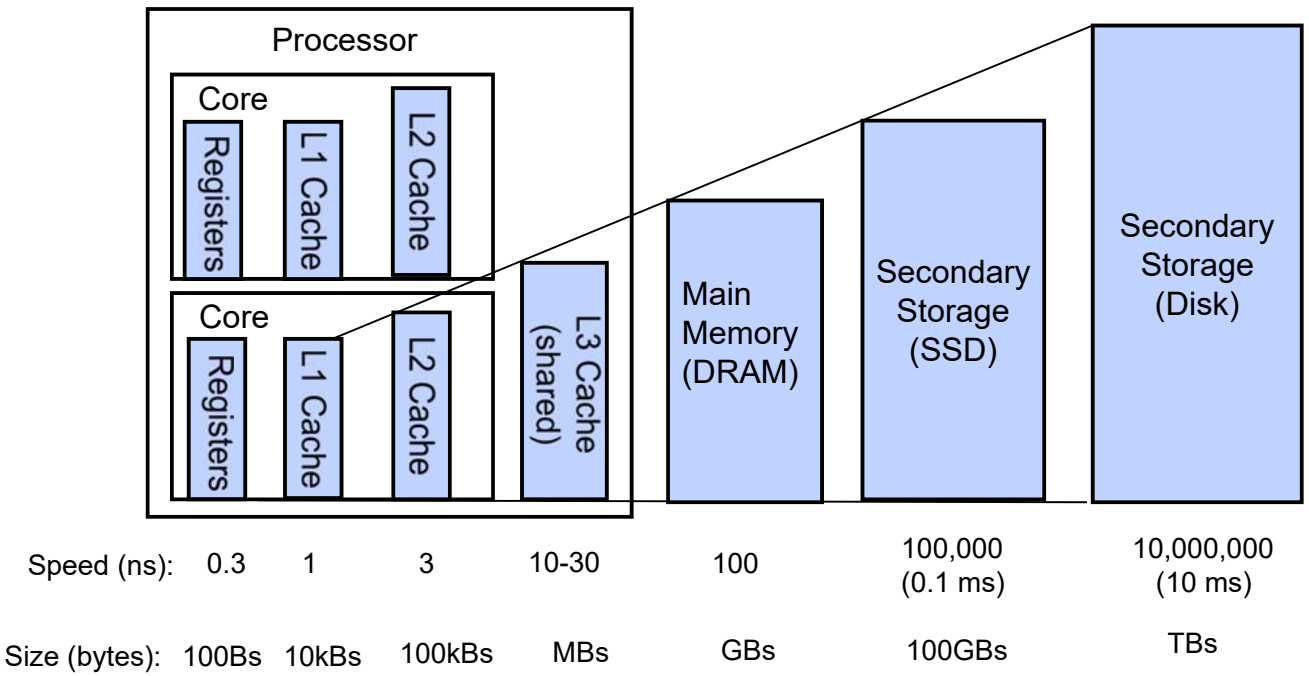
- Idealer Speicher besitzt
  - nahezu unbegrenzte Speicherkapazität
  - geringe Speicherkosten
  - kurze Zugriffszeit bei wahlfreiem Zugriff
  - hohe Zugriffsraten
  - Nichtflüchtigkeit
  - Fähigkeit zu logischen, arithmetischen u.ä. Verknüpfungen
- Speicherhierarchie versucht Annäherung an idealen Speicher durch reale Speichermedien durch Nutzung von Lokalitätseigenschaften zu erreichen
  - Pufferung/Allokation von Daten mit hoher Zugriffswahrscheinlichkeit in schnelle (relativ kleine) Speicher
  - Mehrheit der Daten verbleibt auf langsameren, kostengünstigeren Speichern

Speicherhierarchie (2)

- Rekursives Prinzip
  - kleinere, schnellere und teurere Cache-Speicher werden benutzt, um Daten zwischenzuspeichern, die sich in größeren, langsameren und billigeren Speichern befinden
- Preis-Leistungs-Tradeoff
  - schneller Speicher ist teuer und deshalb klein
  - Speicher hoher Kapazität ist typischerweise langsamer

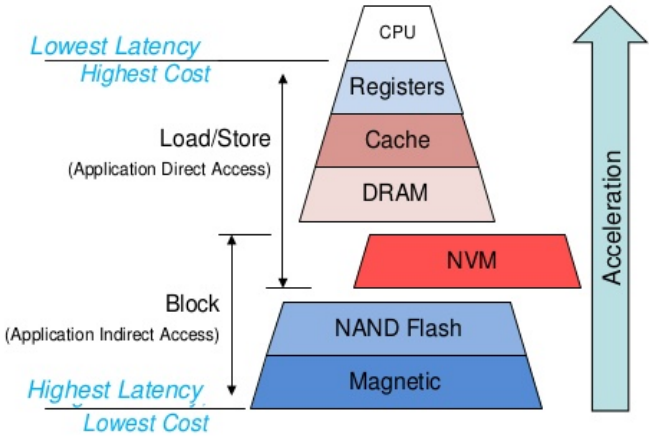


Aktuelle Speicherhierachie



Nichtflüchtiger Halbleiterspeicher

- neuer Speichertyp:
  non-volatile/persistent RAM
  - Bsp,: Intel Optane DC Persistent Memory
  - langsamer/preiswerter als RAM
  - schneller/teurer als SSD
  - kleinerer Zugriffseinheiten
    als Seiten (cache lines)



- kann als sehr schnelle SSD benutzt werden, zB für schnelle Writes/Commits und Recovery-Beschleunigung
- allgemeine Verfügbarkeit erst ab 2019

Speicherhierarchie: Nutzung

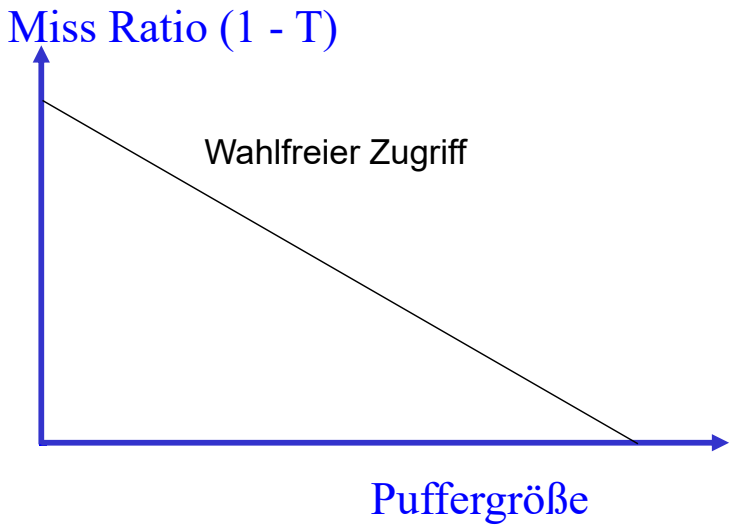
- ähnliche Verwaltungsaufgaben auf jeder Ebene der Speicherhierarchie:
  - Lokalisieren von Datenobjekten
  - Allokation von Speicherplatz
  - Ersetzung
  - Schreibstrategie (Write-through vs. Write-back)
  - ggf. Anpassung an verschiedene Transfergranulate

- warum funktionieren Speicherhierarchien?
  - "perfekter Speicher" würde die Daten, die demnächst angefordert werden, schon bereithalten
  - Annäherung durch Lokalitätsprinzip
  - temporale Lokalität: Pufferung kürzlich referenzierter Objekte
  - räumliche Lokalität: Pufferung benachbart gespeicherter Objekte

Cache-Nutzung

- Cache-Trefferrate

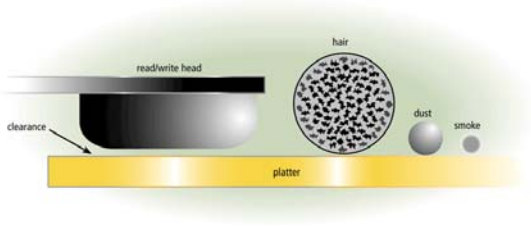
  T = #Treffer im Cache / alle Referenzen auf Cache



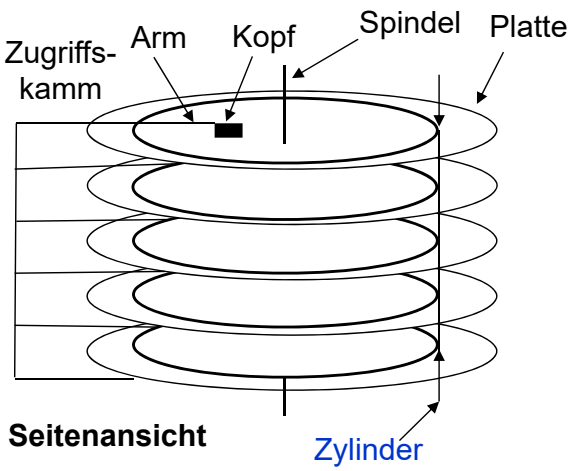
- Verbesserung der Trefferrate
  - größere Caches
  - Clusterbildung für referenzierte Daten (räumliche Lokalität)

Magnetplattenspeicher

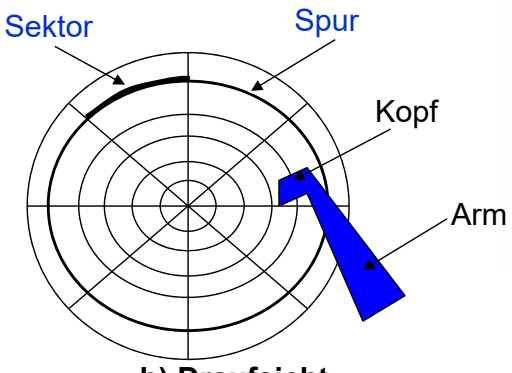
- Aufbau
  - mehrere gleichförmig rotierende Platten mit eigenem Schreib-/Lesekopf
  - jede Plattenoberfläche ist eingeteilt in Spuren
  - die Spuren sind formatiert als Sektoren fester Größe (Slots)
  - Sektoren (z.B. 2 - 32 KB) sind kleinste Schreib-/Leseeinheit
  - jeder Sektor speichert selbstkorrigierende Fehlercodes



- Adressierung
  - Zylindernummer, Spurnummer, Sektornummer



a) **Seitenansicht**



b) Draufsicht

Magnetplatten: Technische Merkmale

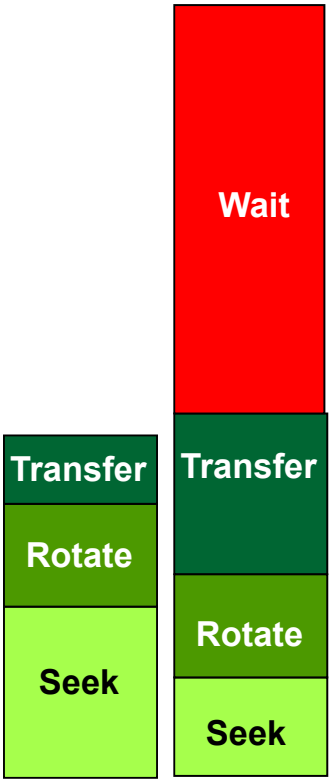
Merkmal \ Magnetplattentyp	typische Werte 2000	IBM 3390 (1990)	IBM 3330 (1970)
t_{smin} = Zugr.bewegung (Min)	0,65 ms	k. A.	10 ms
t_{sav} = " (Mittel)	4,1 ms	12.5 ms	30 ms
t_{smax} = " (Max.)	8,5 ms	k. A.	55 ms
t_{rev} = Umdrehungszeit	4 ms	14.1.ms	16.7 ms
T_{cap} = Spurkapazität	178 KB	56 KB	13KB
T_{cyl} = #Spuren pro Zyl.	11	15	19
N_{dev} = #Zylinder	18700	2226	411
u = Transferrate	45 MB/s	4.2 MB/s	0. 8 MB/s
Nettokapazität	36,7 GB	1.89 GB	0.094 GB

- typische Werte in 2017:
  - 2-8 TB Kapazität, 50 - 120 MB/s (mit Cache-Puffer mehr)
  - 3 - 8 ms Seek, 4-8 ms Umdrehungszeit (7200-15.000 RPM, Rotations per Minute)
  - 20-50 $ / TB im Low-Cost-Sektor (SATA)

Komponenten der Platten-Zugriffszeit

- Vorbereitung des E/A-Zugriffs (SVC) ≈ 2500 Instr.
- stets anfallende Zeitanteile
  - Starten E/A durch Betriebssystem
  - Zugriffsarmbewegung (seek) (t_s)
  - Aktivieren Schreib-/Lesekopf
  - Umdrehungswartezeit (revolve) (t_{rev})
  - Wiederbelegen Kanal
  - Übertragungszeit (transfer) (t_{tr})
  - Prüfzeit
- vereinfachtes Modell für die Zugriffszeit

$$t = t_s + t_r + t_{tr}$$



- zusätzlich: lastabhängige Wartezeiten auf Platte bzw. Übertragung (oft Hauptanteil bei gemeinsam genutzten Platten)

Wahlfreier vs. sequenzieller Zugriff

- nur geringe Verbesserungen bei Zugriffszeit
  (Seek t_{sav}, Umdrehungszeit t_r)
- starke Verbesserungen bei Aufzeichnungsdichte und Kapazität
- wie teuer sind der sequenzielle Zugriff und der wahlfreie Zugriff auf 1000 Blöcke?

$t_{random} = 1000 \times (t_{sav} + t_{rev}/2 + t_{tr})$

$t_{sequenziell} =$

	1970	2015	**Verbesserung**
wahlfrei			
sequenziell			
Verhältnis n			

Wahlfreier vs. sequenzieller Zugriff (2)

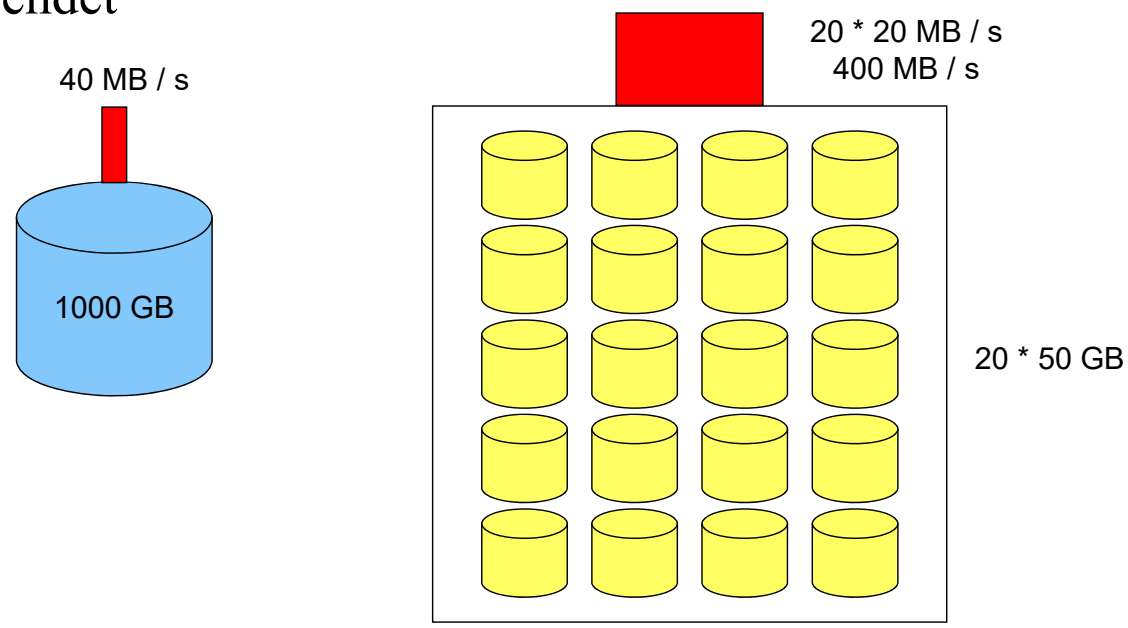
- Beobachtungen
  - sequenzieller Zugriff zu Magnetplatten ist bei großen Datenmengen n-mal schneller als wahlfreier Zugriff zu denselben Daten
  - da Transferraten (u und Kapazität) schneller wachsen als Zugriffsraten, wird das Verhältnis wahlfreier zu sequenzieller Zugriff immer schlechter
- Folgende Maßnahmen werden immer wichtiger
  - große Blöcke: Wahl größerer Transfereinheiten verbessert dieses Verhältnis
  - Clusterbildung der Daten: Datencluster sollen für Zugriff auf Magnetplatten gebildet und bewahrt werden, so dass mit großen Blöcken in möglichst wenigen Zugriffen möglichst viele nützliche Daten übertragen werden

Entwicklungstrends: magnetische Speicher

- starke Verbesserungen bei
  - Lese- und Schreibdichte (> Faktor 100 innerhalb einer Dekade)
  - analoge Verbesserungen in Kapazität sowie Kosten pro GB
- nur geringe Verbesserungen bei
  - Zugriffszeiten (Faktor 2 in 10 Jahren)
  - E/A-Raten (#IO / s)
- Konsequenzen
  - Plattengeschwindigkeit verbessert sich weit geringer als CPU-Geschwindigkeit (Verdoppelung alle 2-3 Jahre)
  - hohe Plattenkapazität kann im Mehrbenutzerbetrieb oft nicht genutzt werden !
  - vollständiges Lesen einer Platte (disk scan), z.B. für Backup, dauert immer länger!
- Trends
  - Disk-Arrays
  - elektronische Zwischenspeicher (Platten-Caches, SSDs), Flash-Speicher
  - Platten ersetzen Bänder als Archivmedium
  - verteilte Online-Archive, z.B. zum Katastrophenschutz

Disk-Arrays

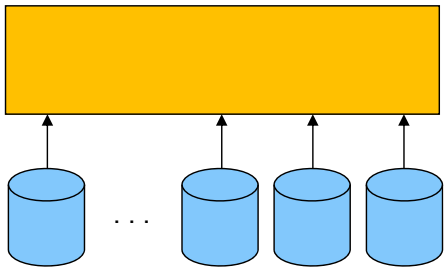
- konventionelle Platten
  - hohe Kapazität
  - relativ geringe Bandbreite und E/A-Rate (ca. 80-200 Zugriffe/s)
- Disk-Arrays: viele kleinere Platten werden logisch als eine Platte verwendet



E/A-Parallelität

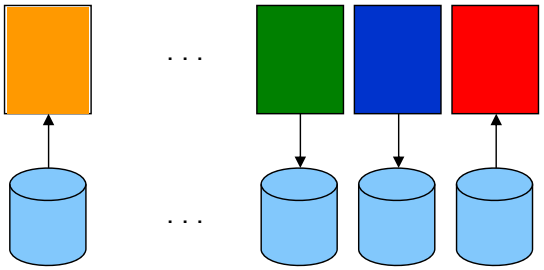
- Voraussetzung: *Declustering* von Dateien über mehrere Platten
- 2 generelle Arten von E/A-Parallelität

***Intra*-E/A-Parallelität**



1 E/A-Auftrag wird in mehrere, parallel ausführbare Plattenzugriffe umgesetzt

***Inter*-E/A-Parallelität**

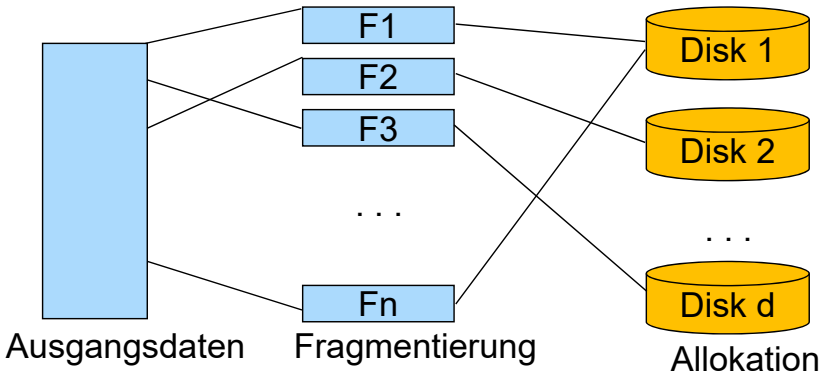


N unabhängige E/A-Aufträge können parallel ausgeführt werden, sofern die betreffenden Daten über verschiedene Platten verteilt sind

- **Probleme**:
  - gleichzeitige Unterstützung von Intra- und Inter-E/A-Parallelität
  - Datenverteilung
  - Fehlertoleranz

Disk-Arrays: Datenverteilung

- Nutzung von E/A-Parallelität setzt Verteilung der Daten (Dateien, Relationen) über mehrere Platten voraus: *Declustering (Partitionierung)*
- Ziele
  - Unterstützung von Inter- und Intra-E/A-Parallelität
  - Lastbalancierung (gleichmäßige Plattenauslastung)

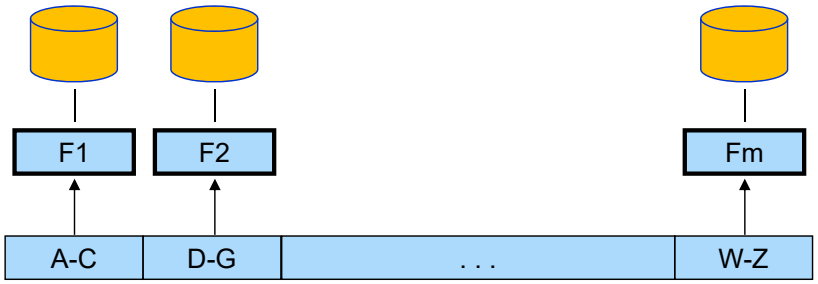


- Teilaufgaben bei der Allokation einer Datei
  - Bestimmung des Verteilgrades d
  - Bestimmung der Verteilgranulate (Fragmente): Logisches vs. physisches Declustering; Umfang / Striping-Granulat (z.B. #Blöcke)
  - Allokation: Zuordnung der Fragmente zu Platten

Logisches Declustering

- Daten werden auf einer anwendungsorientierten Ebene partitioniert
- üblich: Zerlegung von Relationen in Tupelmengen aufgrund der Werte eines (Verteil-) Attributs, z.B. über Hash- oder Bereichspartitionierung

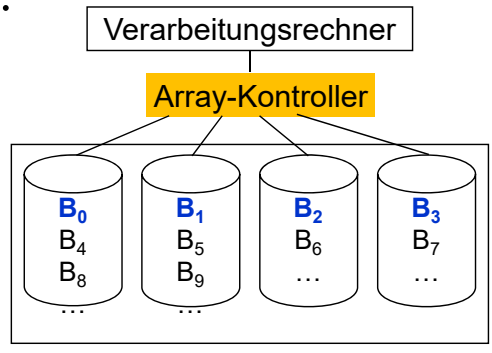
Beispiel:
Verteilattribut NAME

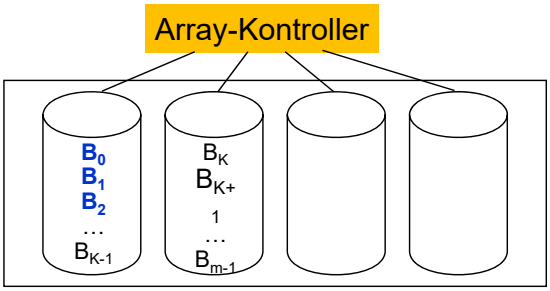


- Festlegung durch DBA (oder Tool)
- Vorteil: DBS kann bekannte Datenverteilung nutzen, logische Operationen auf einer Teilmenge der Daten zu begrenzen
- effektive Parallelisierung von Operationen wird unterstützt, so dass parallele Teiloperationen auf disjunkten Datenmengen / Platten arbeiten können

Physisches Declustering (Striping)

- Realisierung durch BS oder Array-Kontroller => Nutzbarkeit durch verschiedene Anwendungen
- Striping-Granulat S für DB-Anwendungen: Block bzw. Blockmengen
- blockweise Fragmentierung + Allokation :
  - Extrem 1: *Round-Robin* über alle Platten (S=1)
  - Extrem 2: *Clusterung* (S=k für Platten-kapazität k; minimale Plattenanzahl)
- Tradeoffs:
  - Intra- vs. Inter-Transaktionsparallelität
  - "Skew"-Effekte
  - Lastbalancierung
  - sequenzielle Zugriffe



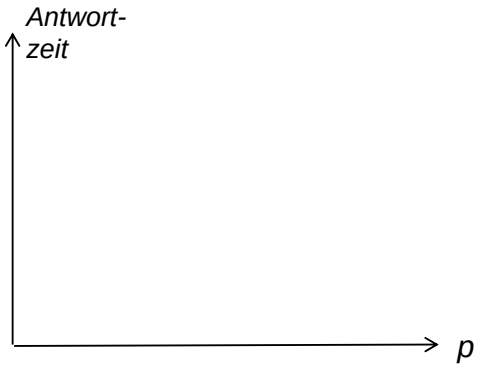


Datenverteilung: Striping-Granulat

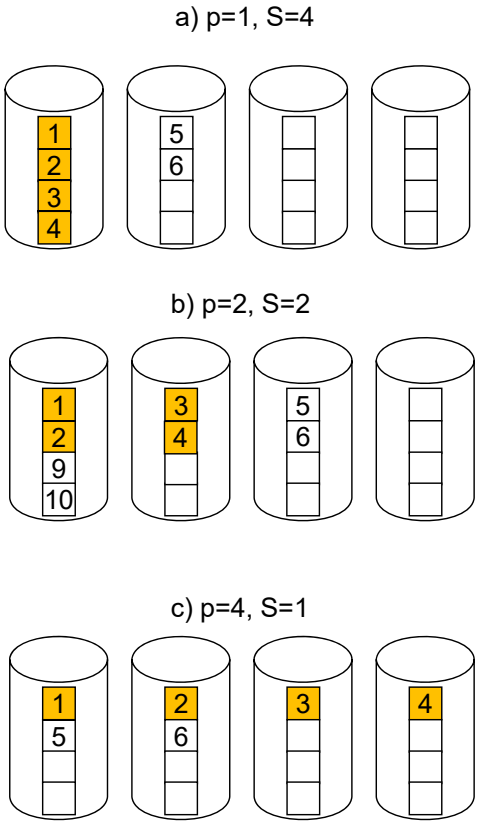
- wünschenswert: dateibezogene Allokation mit Hinblick auf
  - Dateigröße
  - Zugriffshäufigkeiten
  - mittl. Auftragssumfang, etc.
- wesentlicher Schritt: Bestimmung des Striping-Granulats S
  - mittlere Auftragsgröße: R Blöcke
  - S kann aus optimalem Parallelitätsgrad / Plattenanzahl p_{opt} für R Blöcke abgeleitet werden

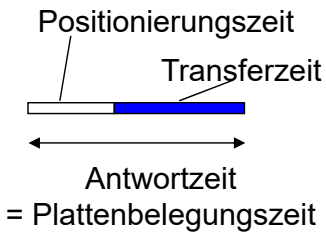
$$\Rightarrow S = \lceil R / p_{opt} \rceil$$

- Bestimmung von p_{opt} durch Abschätzung der Antwortzeit in Abhängigkeit der Plattenanzahl p
  - wachsendes p verbessert lediglich Transferanteil der Zugriffszeit
  - Positionierungszeiten sind dagegen von der langsamsten Platte bestimmt (Verschlechterung mit wachsendem p)
  - Zunahme der Plattenbelegungszeiten auch ungünstig für Durchsatz



Beispiel für R=4





Allokation / Lastbalancierung

- Zielsetzung der Allokation: Lastbalancierung
  - Gesamtlast auf alle Fragmente soll gleichmäßig über alle Platten verteilt sein
- Varianz der „Plattenhitzen“ soll minimal sein
  - *Hitze*: Summe der Zugriffshäufigkeiten aller Blöcke eines Fragments
  - Hitze einer Platte: akkumulierte Hitze ihrer Fragmente
- einfache Fragment-Allokation: Round Robin oder Random
  - garantiert keine gute Lastbalancierung bei bei ungleichmäßiger Hitzeverteilung in den Fragmenten (z.B. 80-20- oder 90-10-Regel bei der Zugriffsverteilung)
- (Greedy) Heuristik zur Lastbalancierung
  - Voraussetzung: Hitze aller Fragmente a priori bekannt
  - Initialisierung: Setze die Gesamthitze aller Platten auf 0
  - Schritt 1: Sortiere die zu allozierenden Fragmente nach absteigender Hitze
  - Schritt 2: Für jedes Fragment in Sortierreihenfolge:
    - alloziere Platz auf der derjenigen Platte mit der geringsten akkumulierten Hitze, die noch genügend freien Platz hat und auf der noch kein Fragment derselben Datei liegt
    - addiere die Hitze des allozierten Fragments zur Gesamthitze der ausgewählten Platte

Beispiel (Lastbalancierung)

Datei 1

	1.1	1.2	1.3	1.4	1.5	1.6
Hitze:	10	4	4	3	2	1

Datei 2

	2.1	2.2	2.3	2.4
Hitze:	8	5	5	1

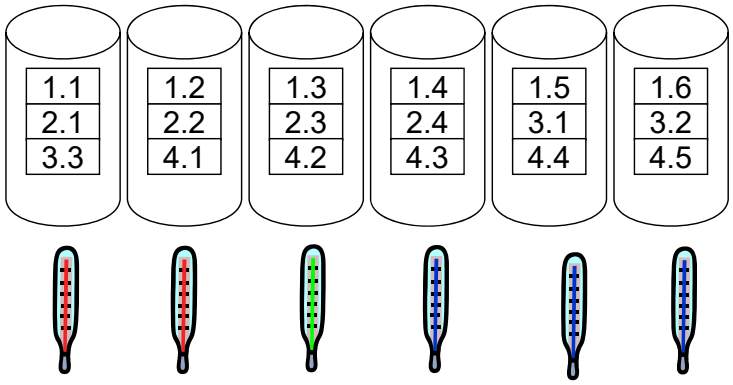
Datei 3

	3.1	3.2	3.3
Hitze:	5	5	5

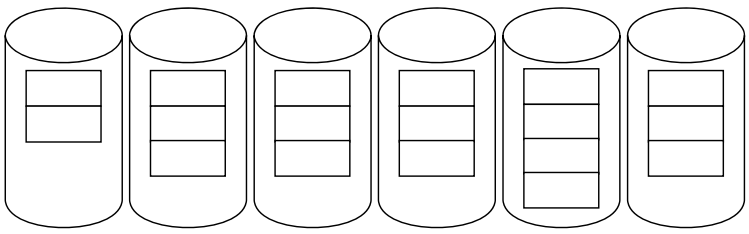
Datei 4

	4.1	4.2	4.3	4.4	4.5
Hitze:	7	4	3	2	1

Round-Robin-Allokation



Greedy-Heuristik



Fehlertoleranz

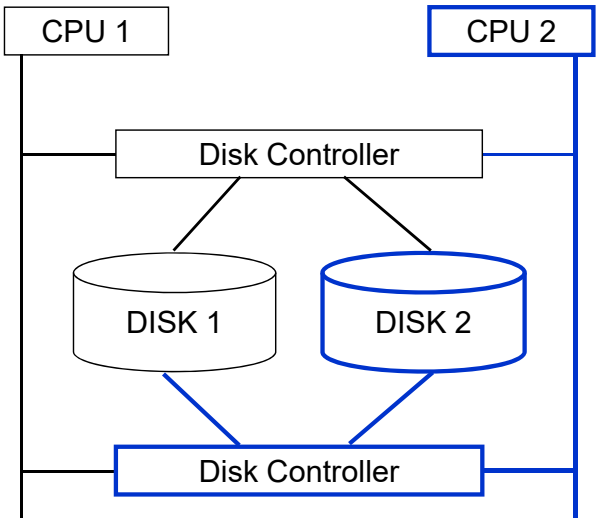
"The Problem with Many Small Disks: Many Small Faults"

- Disk-Array mit N Platten: ohne Fehlertoleranzmechanismen N-fach erhöhte Ausfallwahrscheinlichkeit => System ist unbrauchbar
- Begriffe
  - Mean Time To Failure (MTTF): Erwartungswert für die Zeit (von der Inbetriebnahme) bis zum Ausfall einer Platte
  - Mean Time To Repair (MTTR): Erwartungswert für die Zeit zur Ersetzung der Platte und der Rekonstruktion der Daten
  - Mean Time To Data Loss (MTTDL): Erwartungswert für die Zeit bis zu einem nicht-maskierbaren Fehler
- Disk-Array mit N Platten ohne Fehlertoleranzmechanismen:

  MTTDL = MTTF / N
- Schlüssel zur Fehlertoleranz ist Redundanz => Redundant Arrays of Independent Disks (RAID)
  - durch Replikation der Daten (z. B. Spiegelplatten) - RAID1
  - durch zusätzlich zu den Daten gespeicherte Error-Correcting-Codes (ECCs), z.B. Paritätsbits (RAID-4, RAID-5)

Spiegelplatten (RAID1)

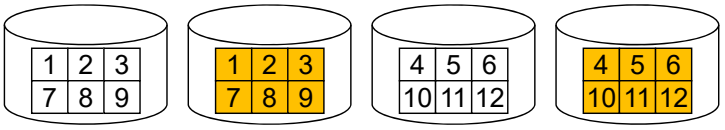
- weit verbreitet zur automatischen Behebung von Plattenfehlern
- Vorteile
  - sehr hohe Verfügbarkeit
  - Optimierung von Lesezugriffen (Lastbalancierung, Minimierung der Positionierungszeiten)
- Nachteile
  - Verdoppelung des Speicherbedarfs
  - Schreibzugriffe zweifach auszuführen
  - Lastbalancierung im Fehlerfall ungünstig



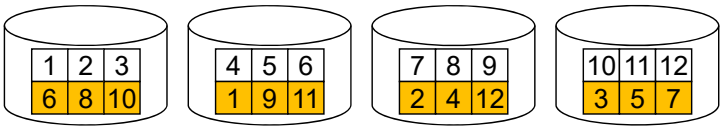
Spiegelplatten (2)

- Verbesserung der Lastbalancierung im Fehlerfall durch verstreute Replikation
  - Verteilung der Kopien zu Daten einer Platte über mehrere andere Platten

a) **Konventionelle Spiegelplatten**



b) **Verstreute Replikation: Interleaved Declustering**

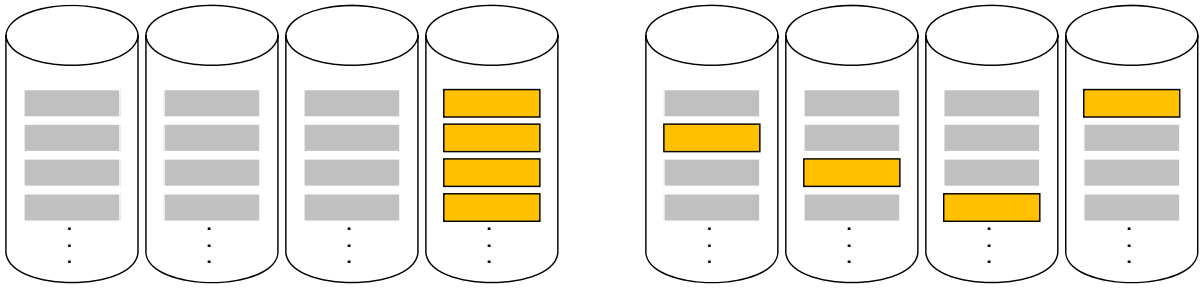


- Gruppenbildung erlaubt flexiblen Kompromiss zwischen Lastbalancierung und Verfügbarkeit
  - Verteilung der Replikate einer Platte nur über G-1 Platten derselben Gruppe
  - Mehrfachfehler führen nicht zu Datenverlust solange verschiedene Gruppen betroffen sind
  - konventionelle Spiegelplatten ergeben sich als Spezialfall mit G=2

Einsatz von Paritätsblöcken (ECC)

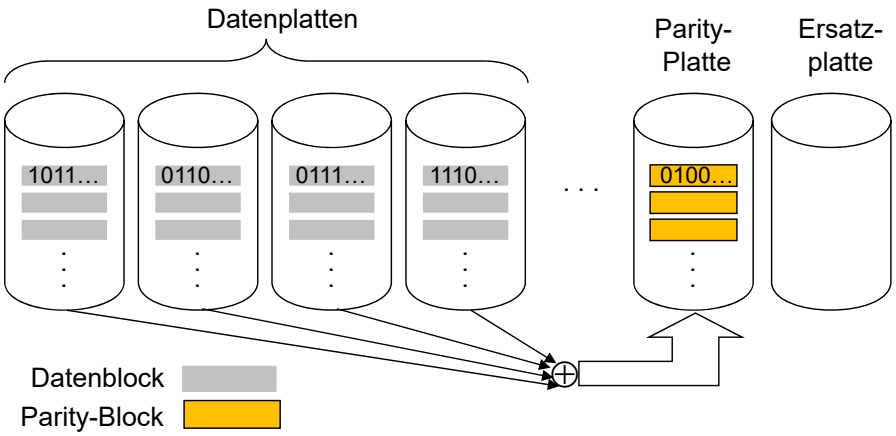
- Bildung von Gruppen mit je G Platten
  - *Paritätsgruppe*: G-1 Datenblöcke + zugehöriger Paritätsblock
  - Erhöhung des Platzbedarfs um lediglich 100/(G-1) %
  - Paritätsblock entspricht EXOR-Bildung von G-1 Datenblöcken
  - bei jeder Änderung eines Datenblockes anzupassen
- RAID-4 vs. RAID-5

  RAID-5 vermeidet Engpass einer einzigen Parity-Platte (“Parity Striping”)



Datenblock Parity-Block

Aktualisierung der Paritätsblöcke

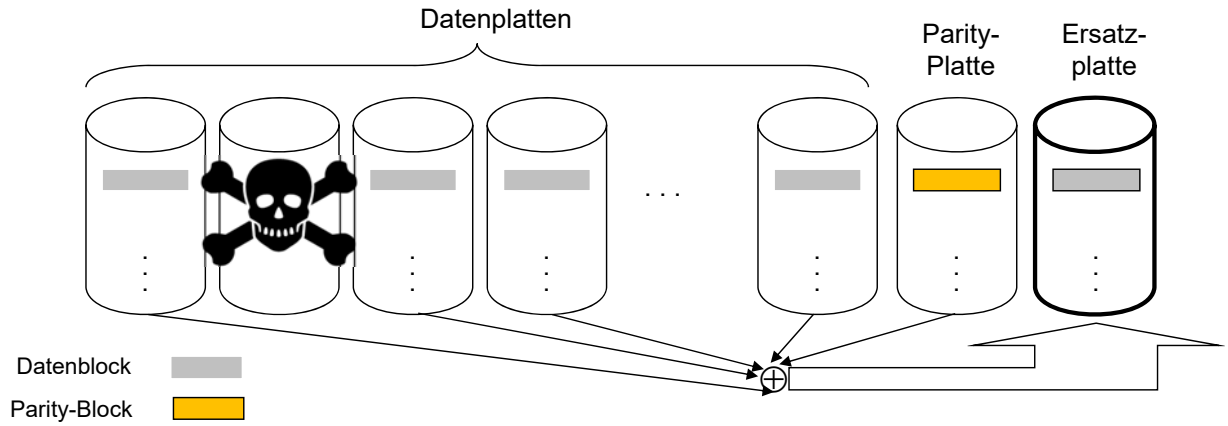


Exor-Operation:
$x_1 \oplus x_2 \oplus \ldots \oplus x_k = 1$
falls Zahl der x_i mit Wert 1 ungerade ist, sonst ist das Ergebnis 0

- inkrementelle Berechnung des Parity-Blocks
  1. Alter Datenblock b und alter Parity-Block p werden parallel in den Speicher des Disk-Array-Controllers (bzw. HSP) gelesen
  2. $p \oplus b \oplus b' = p'$
  3. Geänderter Datenblock b' und neu-berechneter Parity-Block p' werden parallel auf Platte zurückgeschrieben
     => *4 Plattenzugriffe* (falls b bereits gepuffert, 3 Zugriffe)
  - relativer Aufwand ist geringer, wenn mehrere Blöcke einer Gruppe geändert werden

Datenrecovery: Rekonstruktion von Blöcken

- Rekonstruktion von Blöcken
  1. Paralleles Lesen des Parity-Blocks p und der Blöcke b_j aus der Gruppe des zu rekonstruierenden Blocks von allen noch verfügbaren Datenplatten
  2. Rekonstruktion des Blockes b_i :
     $(b_1 \oplus \ldots \oplus b_{i-1} \oplus b_{i+1} \oplus \ldots \oplus b_{G-1}) \oplus p =$
     $(b_1 \oplus \ldots \oplus b_{i-1} \oplus b_{i+1} \oplus \ldots \oplus b_{G-1}) \oplus (b_1 \oplus \ldots \oplus b_{G-1}) = b_i$
  3. Zurückschreiben von b_i auf die Ersatzplatte

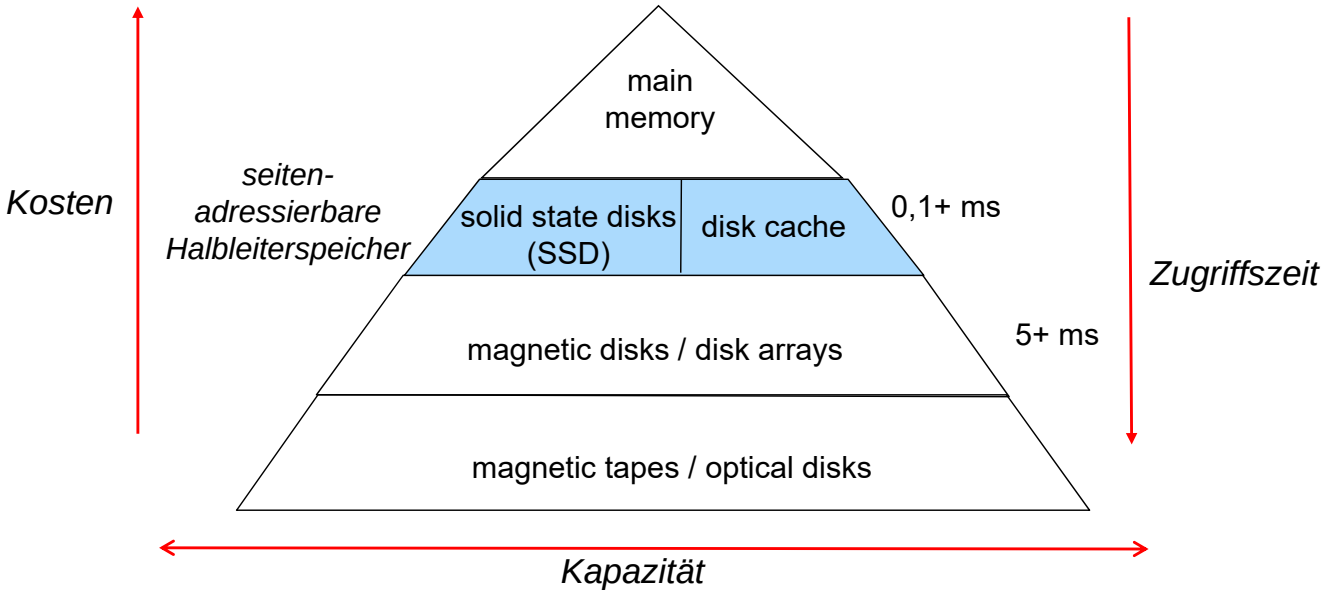


RAID-1 vs. RAID-5

Merkmal	RAID-0 (keine Redundanz)	RAID-1	RAID-5
Speichereffizienz (Kosteneffektivität)	1	**1 / 2**	**(G-1) / G**
Datenverfügbarkeit	--		
„Large Reads“ (Datentransferrate für Lesen)	1		
„Large Writes“ (Datentransferrate)	1		
„Small Reads“(E/A-Rate)	1		
„Small Writes“(E/A-Rate)	1		

- spezifische RAID-1-Vorteile: gute Lesezugriffszeiten und schnelle Platten-Recovery

Erweiterte Speicherhierarchie



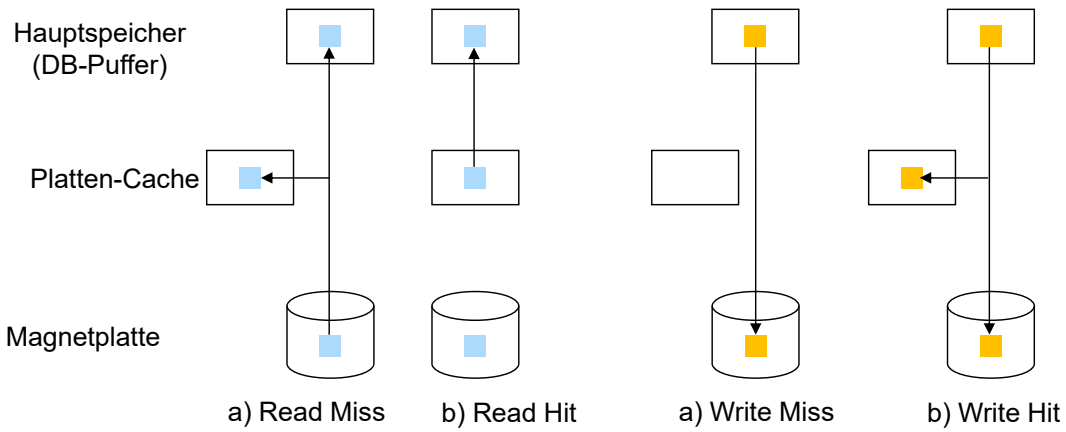
- Platten-Cache und SSD:
  Verwaltung durch eigene Kontroller, allgemeine Plattenschnittstelle
- Nicht-Flüchtigkeit: Battery-Backup / ununterbrechbare Stromversorgung

Einsatzformen seitenadressierbarer Halbleiterspeicher

- **Pufferung von DB-Seiten** auf mehreren Speicherebenen
  - Einsparung von Lesezugriffen auf Platte
  - keine Nicht-Flüchtigkeit erforderlich
  - realisierbar mit Platten-Cache
- **Schreibpuffer** in nicht-flüchtigem Halbleiterspeicher (z.B. Platten-Cache)
  - synchrones Schreiben in Schreibpuffer, asynchrones Durchschreiben auf Platte
  - Antwortzeitverbesserung bereits durch kleinen Pufferbereich
- **Speicherung ganzer Dateien** in nicht-flüchtigen Halbleiterspeichern wie SSD
  - Vermeidung aller Plattenzugriffe
  - hohe Kosten

Einsatz von Platten-Caches

- Flüchtige Platten-Caches: nur Verbesserung für Lesezugriffe
  - Problem: Doppel-Pufferung im Hauptspeicher und Platten-Cache



- Prefetching für sequenzielle Dateien
- nicht-flüchtiger Schreibpuffer im Platten-Cache
  - starke Verbesserung von Schreibzugriffen (besonders wirksam für RAID-5)
  - bessere Antwortzeiten
  - höhere Plattenauslastung tolerierbar

Flash-Speicher

- rapide Verbreitung in mobilen Geräten (MP3-Player, Kameras, Notebooks…)
- nur noch moderat teurer als Magnetplatten (Faktor 4-10), Unterschied wird geringer
  - leise, geringer Energieverbrauch (da keine mechanische Bewegungen)
  - robust gegen Erschütterungen
- sehr schnelle Lesezugriffe (ca 0,1 ms / Seite; gleichmäßige Zeiten; sehr hohe Leseraten)
- langsame Schreibzugriffe (v.a. für NAND-Flash-Chips)
  - erst Löschen eines kompletten Bereichs (z.B. 64 Seiten /128 KB), dann neu Schreiben

Vergleichswerte SSD-HDD-DRAM (2018)

	Bandbreite (sequential R/W)	Kosten/GB	Größe
HDD	50-120 MB/s	$0,03/GB	2-8 TB
SSD*	200-550 MB/s (SATA) 6 GB/s (PCI)	$0,2/GB	200GB-2TB
DRAM	15-25 GB/s	$2-5/GB	128GB-1TB

Bandbreite: SSD >5x höher als HDD, DRAM > 10x als SSD
Kosten: HDD x5 geringer als SSD, SSD x8 weniger als DRAM

Speicherallokation von Dateien

- wo soll Datei gespeichert werden ?
  - Platte
  - nicht-flüchtiger Halbleiterspeicher (z.B. Flash-SSD)
  - Hauptspeicher (-> Hauptspeicherdatenbank)
- Entscheidung abhängig von Speichereigenschaften (Kosten, Zugriffsgeschwindigkeit) und Zugriffsmerkmalen
- HS (bzw. SSD)-Allokation
  - spätestens, wenn HS- bzw. SSD-Kosten unter Plattenkosten
  - für Daten mit hoher Zugriffshäufigkeit bzw. wenn Kosten der Plattenallokation von E/A-Rate anstatt von Speicherkosten bestimmt sind

5-Minuten-Regel (Jim Gray)

- Gray1987/Gray1997: ein Block, auf den spätestens alle 5 Minuten zugegriffen wird, soll im Hauptspeicher bleiben
- Parameter bezüglich Technologie und Kosten
  - Kosten pro Plattenzugriff *(PricePerDrive / AccessPerSecondPerDisk)*
  - Kosten pro HS-Seite: *(PricePerMB / PagesPerMB)*
- Zugriffsfrequenz pro Block (Hitze) f
- Zeitabstand zwischen Zugriffen T = 1/f
- Hauptspeicherspeicherung günstiger wenn gilt

Beispielwerte	1987	1997	2007	2017
PricePer DiskDrive($)	30000	2000	80	49
ApsPerDisk	15	64	83	200
PagesPerMB	1000	128	16	16
PricePerMB ($)	5000	15	0,05	0,005

$$f \times \frac{PricePerDiskDrive}{AccessPerSecondPerDisk} >= \frac{PricePerMB}{PagesPerMB}$$

$$T <= \frac{PagesPerMB}{AccessPerSecondPerDisk} \times \frac{PricePerDiskDrive}{PricePerMB}$$

- Änderung bezüglich T nur, wenn sich Verhältnisse ändern

SSD-Nutzung

Metric	DRAM				HDD				SATA Flash SSD	
	1987	1997	2007	2017	1987	1997	2007	2017	2007	2017
Unit price($)	5k	15k	48	80	30k	2k	80	49	1k	560
Unit capacity	1MB	1GB	1GB	16GB	180MB	9GB	250GB	2TB	32GB	800GB
$/MB	5k	14.6	0.05	0.005	83.33	0.22	0.0003	0.00002	0.03	0.0007
Random IOPS	-	-	-	-	15	64	83	200	6.2k	67k (r)/20k (w)
Sequential b/w (MB/s)	-	-	-	-	1	10	300	200	66	500 (r)/460 (w)

Table 1: The evolution of DRAM, HDD, and Flash SSD properties

Appuswamy, Raja, et al. *The five minute rule thirty years later and its impact on the storage hierarchy*. 2017

- DRAM-HDD: jetzt 2 Stunden-Regel für 8 KB-Seiten und 2-stufige Hierarchie
- DRAM-SSD: 3,5 Min. für Reads / 12 Min. für Writes (8 KB-Seiten)
- SSD – HDD: 12 Stunden (8 KB-Seiten)
  - nur noch „kalte“ Daten auf HDD

Zusammenfassung

- Speicherhierarchie:
  - Geschwindigkeitsanpassung größerer / langsamerer Speicher an die schnellerer zu vertretbaren Kosten (Kosteneffektivität)
- lange Zeit dominierender Speichertyp als Externspeicher: Magnetplatte
  - hohe Steigerung der Speicherkapazität, geringe Kosten
  - jedoch nur geringfügige Verbesserung der Zugriffszeit
  - wachsende Diskrepanz zwischen sequenzieller und wahlfreier E/A-Leistung
  - zunehmende Dauer zum vollständigen Lesen einer ganzen Platte (z.B. für Backup)
- Disk-Arrays: Leistungsverbesserung durch E/A-Parallelität
  - E/A-Rate (Auftragsparallelität) und Bandbreite (Zugriffsparallelität)
  - geeignete Datenverteilung (Declustering) erforderlich
  - automatische Behandlung von Plattenfehlern obligatorisch

Zusammenfassung (2)

- Datenreplikation (Spiegelplatten)
  - hohe Speicherkosten vs. sehr hoher Verfügbarkeit und guter Leistungsfähigkeit
  - Unterstützung der Lastbalancierung im Fehlerfall durch verstreute Replikation
- Nutzung von Paritätsblöcken (RAID-5)
  - geringer Mehrbedarf an Speicherplatz (z.B. 10%)
  - teure Schreibzugriffe (bis zu 4 Plattenzugriffe) -> Nutzung nicht-flüchtiger Schreibpuffer
  - schlechtes Leistungsverhalten im Fehlerfall
- Schließung der Zugriffslücke durch nicht-flüchtige Halbleiterspeicher/Flash-SSDs
  - fallende SSD-Speicherkosten führen zur zunehmender Nutzung an Stelle von Magnetplatten
  - Magnetplatten nur noch für weniger leistungskritische Einsatzfälle / „kalte“ Daten