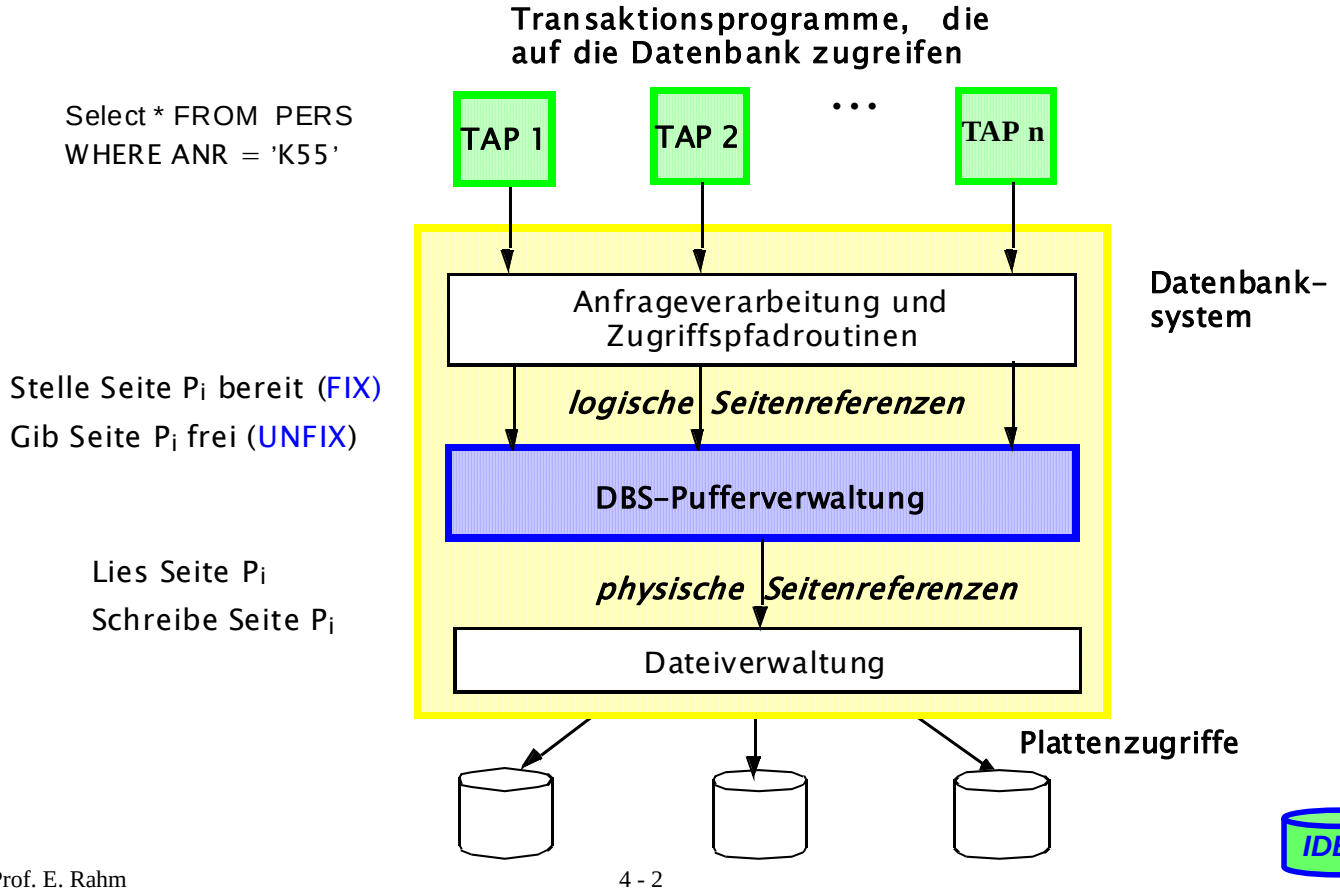
4. DBS-Pufferverwaltung

- Allgemeine Charakteristika
  - Ablauf des Pufferzugriffs
  - Referenzstrings, Stacktiefenverteilung
- Speicherzuteilung im Puffer
- Suche im Puffer
- Schreibstrategien (Force vs. Noforce)
- Lesestrategien (Prefetching, Demand Fetching)
- Seitenersetzungsverfahren
  - Klassifikation von Ersetzungsverfahren
  - LRU, FIFO, CLOCK, GCLOCK, LRD ...
  - Leistungsanalyse von Ersetzungsverfahren
- Neuere Ersetzungsverfahren
  - LRU-K
  - Prioritätsgesteuerte Seitenersetzung (LRU-Priority)
  - Adaptives LRU

IDBS

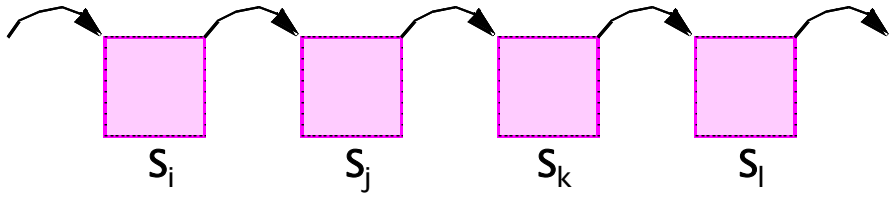
Stellung der Pufferverwaltung innerhalb eines DBS



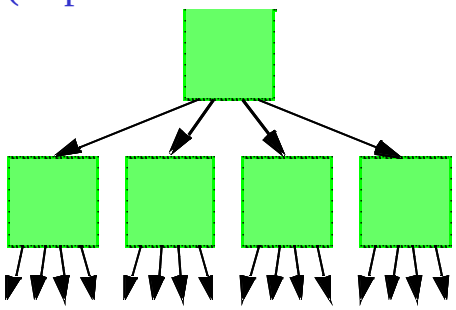
IDBS

Typische Referenzmuster in DBS

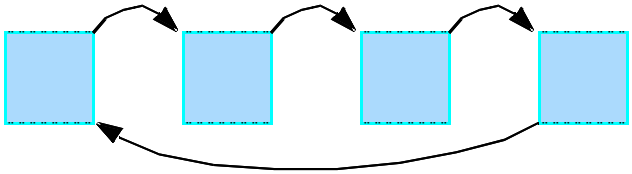
1. Sequentielle Suche (Bsp.: Relationen-Scan)



2. Hierarchische Pfade (Bsp.: Suchen über B*-Bäume)



3. Zyklische Pfade (Bsp.: Abarbeitung verketteter Satzmengen)



Seitenreferenzstrings

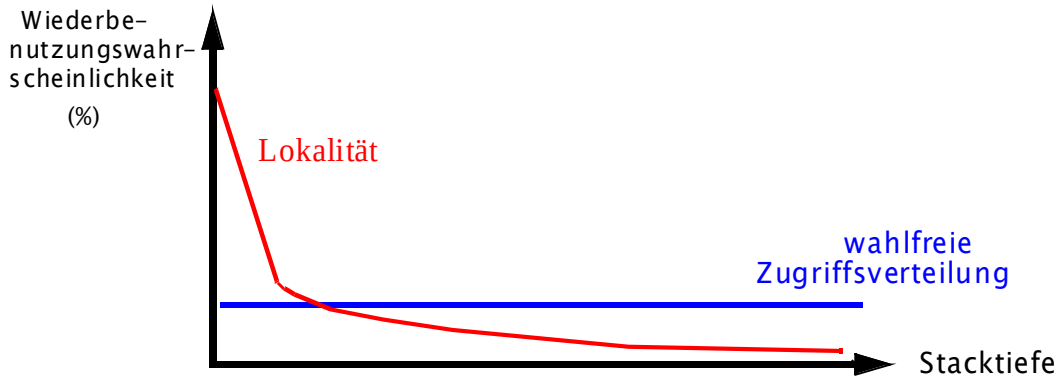
- jede Datenanforderung ist eine *logische Seitenreferenz*
- Aufgabe der Pufferverwaltung: Minimierung der *physischen Seitenreferenzen*
- Referenzstring $R = <r_1, r_2, \ldots r_i, \ldots r_n>$ mit $r_i = (T_i, D_i, S_i)$

  T_i zugreifende Transaktion
  D_i referenzierte DB-Partition
  S_i referenzierte DB-Seite

- Referenzstring-Information ermöglicht
  - Charakterisierung des Referenzverhaltens
    - insgesamt
    - bezüglich bestimmter Transaktionen, Transaktions-Typen und DB-Partitionen
  - Bestimmung von Lokalität und Sequentialität

LRU-Stacktiefenverteilung

- Maß für die Lokalität
- LRU-Stack enthält alle bereits referenzierten Seiten in der Reihenfolge ihres Zugriffsalters
- Bestimmung der Stacktiefenverteilung:
  - pro Stackposition i wird Zähler c_i geführt
  - Rereferenz einer Seite führt zur Zählererhöhung für die jeweilige Stackposition => Zählerwerte entsprechen der Wiederbenutzungshäufigkeit
  - Für LRU-Seitenersetzung kann aus der Stacktiefenverteilung für eine bestimmte Puffergröße unmittelbar die Trefferrate (bzw. Fehlseitenrate) bestimmt werden

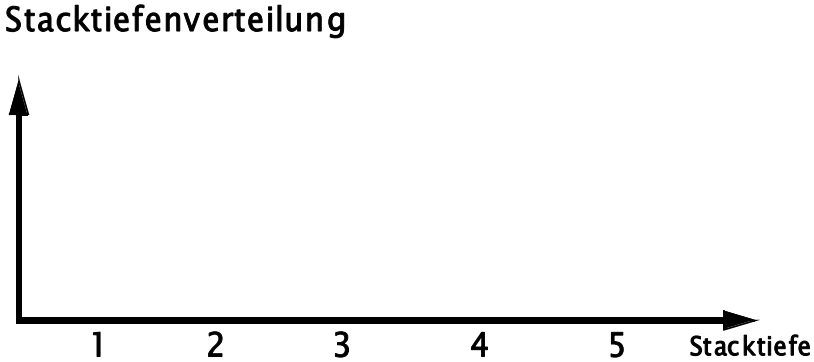


Beispiel

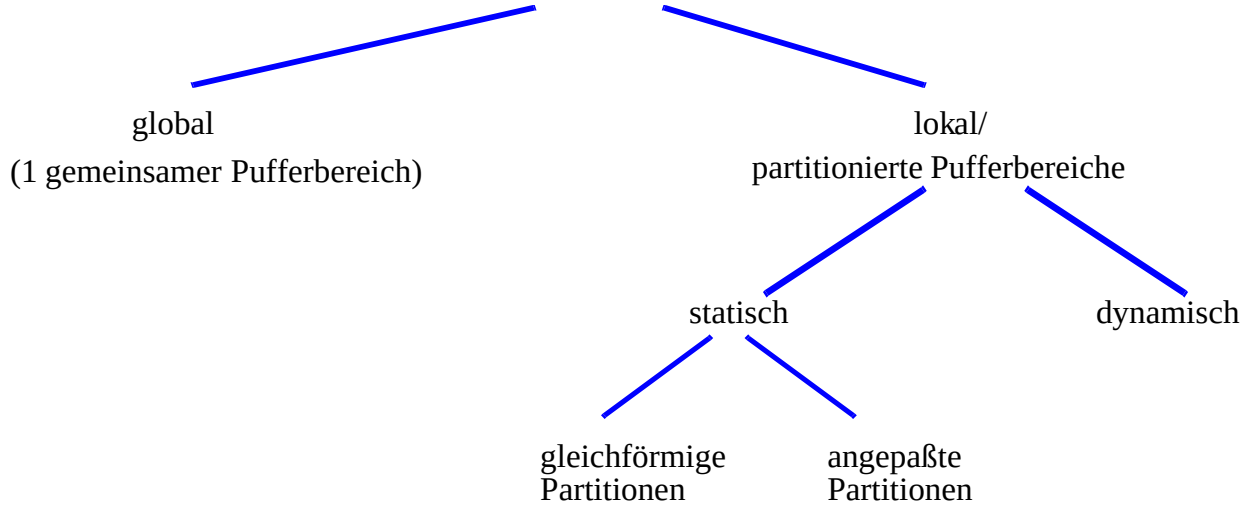
Referenzstring: A B A C C B C D E E A

LRU-Stack:

1	A
2	B
3	C
4	D
5	E



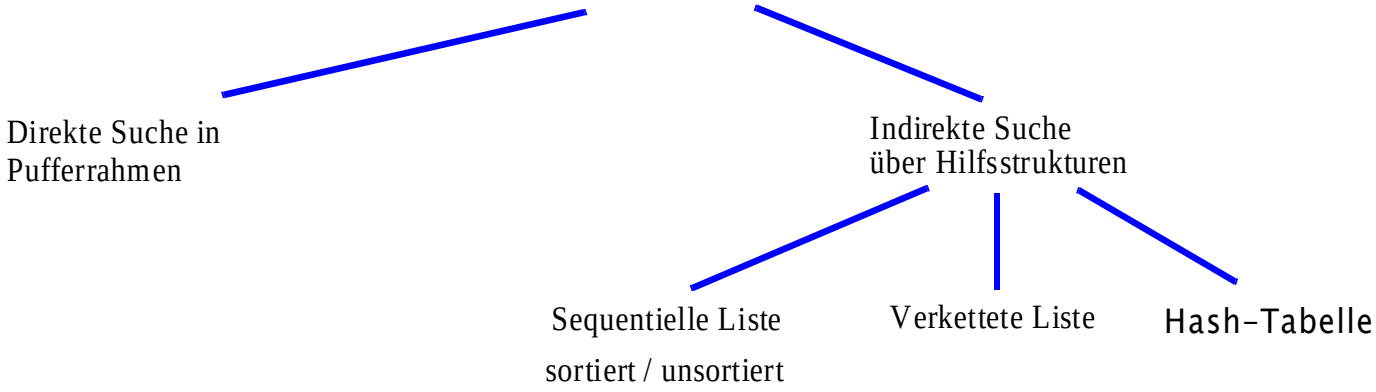
Speicherzuteilung im DB-Puffer



Partitionierungsmöglichkeiten:

- eigener Pufferbereich pro Transaktion bzw. Query
- Transaktionstyp-bezogene Pufferbereiche
- Seitentyp-bezogene Pufferbereiche
- DB (-Partitions)-spezifische Pufferbereiche

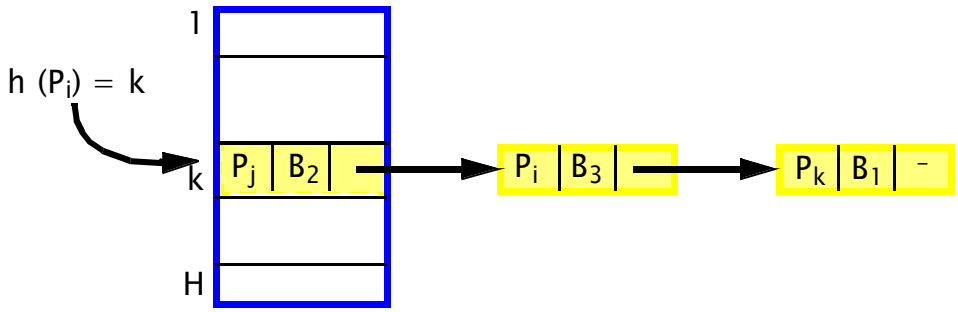
Suche im Puffer



- Probleme der direkten Suche
  - hohe lineare Suchkosten proportional zur Puffergröße
  - hohe Wahrscheinlichkeit von Paging-I/Os
- Listenstrukturen: lineare Suchkosten
- Beste Lösung: Hash-Tabelle mit Kollisionsbehandlung
  - z.B. durch Überlaufketten

Suche (2)

- Hash-Tabelle mit Überlaufketten
- Infos pro Eintrag



  - Seiten-Nummer
  - Pufferadresse
  - Fix-Zähler
  - Änderungsbit (bzw. Änderungszähler)
  - evtl. Zeitpunkt der ersten Änderung etc.

IDBS

Schreibstrategien

- Ersetzung einer geänderten Seite erfordert vorheriges Zurückschreiben der Änderung in permanente DB auf Externspeicher
  - synchrones (=> Antwortzeitverschlechterung) vs. asynchrones Ausschreiben
  - Abhängigkeit zur gewählten Ausschreibstrategie (Force vs. Noforce)
- FORCE: alle Änderungen einer Transaktion werden spätestens beim Commit in die DB zurückgeschrieben ("write-through")
  - i.a. stets ungeänderte Seiten zur Ersetzung vorhanden
  - vereinfachte Recovery nach Rechnerausfall (Änderungen beendeter Transaktionen sind bereits in der permanenten DB)
  - hoher E/A-Overhead
  - starke Antwortzeiterhöhung für Änderungstransaktionen
- NOFORCE: kein Durchschreiben der Änderungen bei Commit (verzögertes Ausschreiben,"deferred write-back")
  - Seite kann mehrfach geändert werden, bevor ein Ausschreiben erfolgt (geringerer E/A-Overhead, bessere Antwortzeiten)
  - vorausschauendes (asynchrones) Ausschreiben geänderter Seiten erlaubt auch bei NOFORCE, vorwiegend ungeänderte Seiten zu ersetzen

  → synchrone Schreibvorgänge in die DB können weitgehend vermieden werden

IDBS

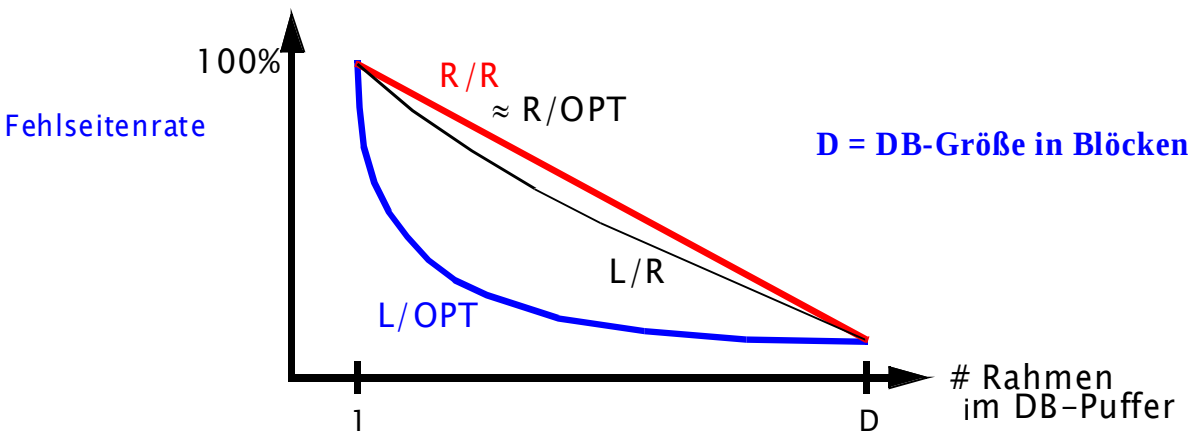
Lesestrategien

Preplanning	Prefetching	Demand Fetching
Transaktionsprogramme; Programmanalyse	physische Datenstrukturierung; Clusterbildung; Verarbeitungswissen	keine Vorausaktionen
große Fehlrate ungenaue Obermengen	falsche Entscheidungen möglich	Lokalitätserhaltung im Puffer

IDBS

Referenzverhalten und Ersetzungsverfahren

- Grundannahme bei Ersetzungsverfahren
  - Refererenzverhalten der jüngeren Vergangenheit ähnelt Referenzverhalten in der näheren Zukunft
  - Nutzung der typischerweise hohen Lokalität bei Ersetzung
- **manchmal** Sequentialität oder zufällige Arbeitslast (RANDOM-Referenzen)
- Kombinationen bzgl. Referenzen/Ersetzung: RANDOM/RANDOM, RANDOM/OPT, Lokalität/RANDOM, Lokalität/OPT
- Grenzfälle zeigen Optimierungsmöglichkeiten auf



IDBS

Kriterien für die Auswahl der zu ersetzenden Pufferseite

Verfahren	Alter	Letzte Referenz	Referenzhäufigkeit	Andere Kriterien
OPT				
RANDOM				
LRU				
LFU				
FIFO				
CLOCK				
GCLOCK				
LRD V1				
LRD V2				
LRU-K				
LRU-Priority				
Adaptives LRU				

IDBS

Least-Recently-Used (LRU)

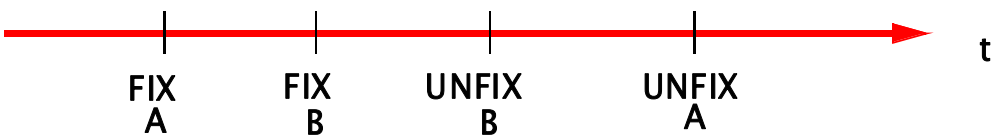
Beispiel (Puffergröße 4):

Referenz der Seite C

LRU-Stack

Referenz der Seite E

Unterscheidung zwischen
Least-Recently-Referenced und
Least-Recently-Unfixed



IDBS

Least-Frequently-Used (LFU)

- Führen eines Referenzzählers pro Seite im Puffer
- Ersetzung der Seite mit der geringsten Referenzhäufigkeit

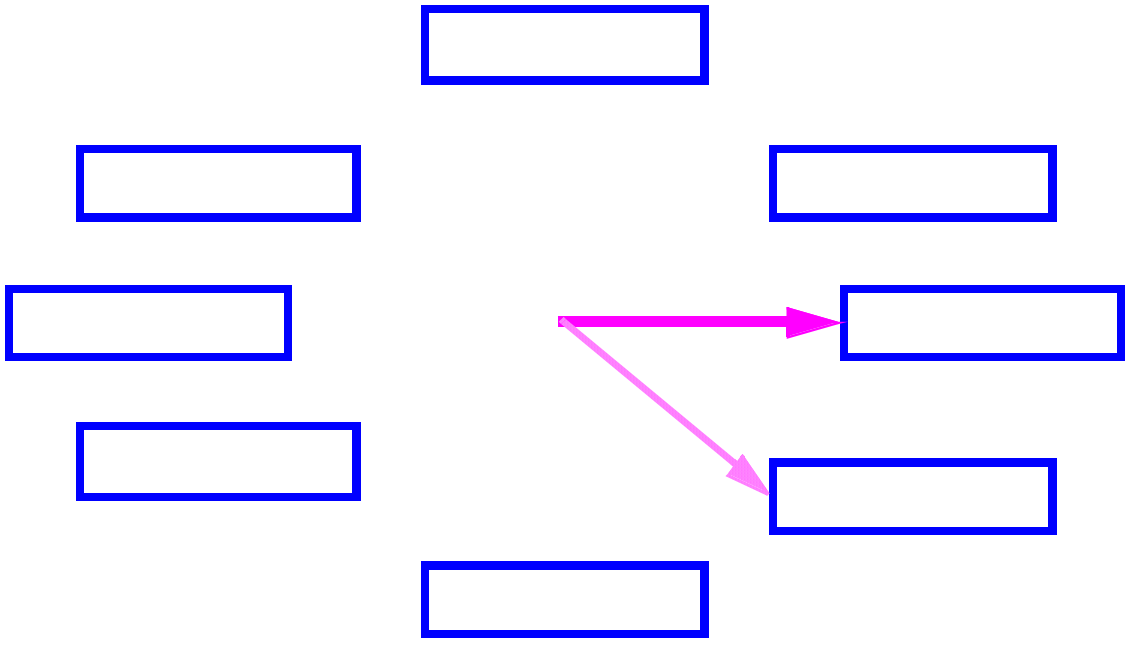
RZ	
2	
4	
1	
3	
3	
6	
1	
3	

- Alter einer Seite wird nicht berücksichtigt

IDBS

FIFO (First-In First-Out)

- die älteste Seite im Puffer wird ersetzt

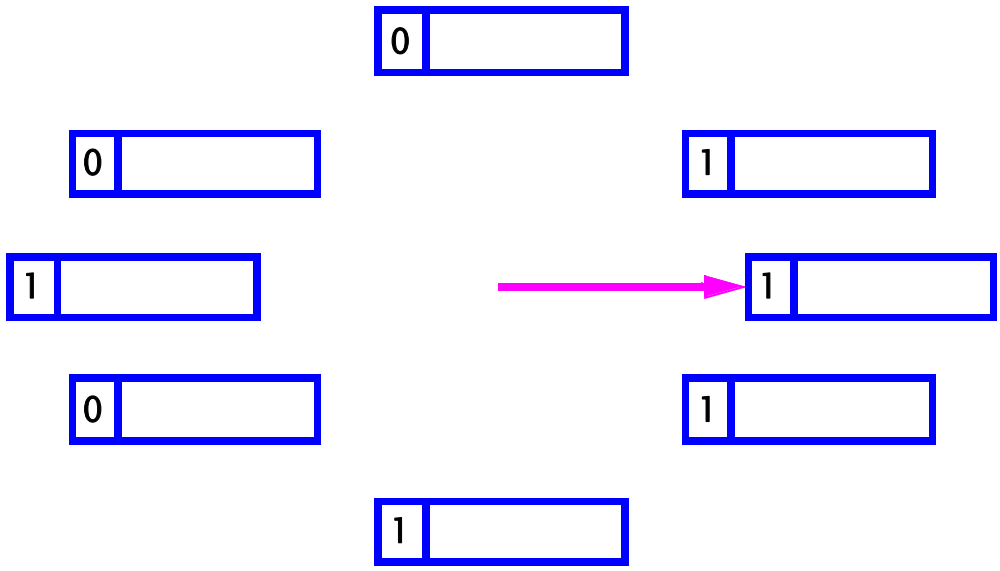


- Referenzierungsverhalten während Pufferaufenthaltes wird nicht berücksichtigt

IDBS

CLOCK (Second Chance)

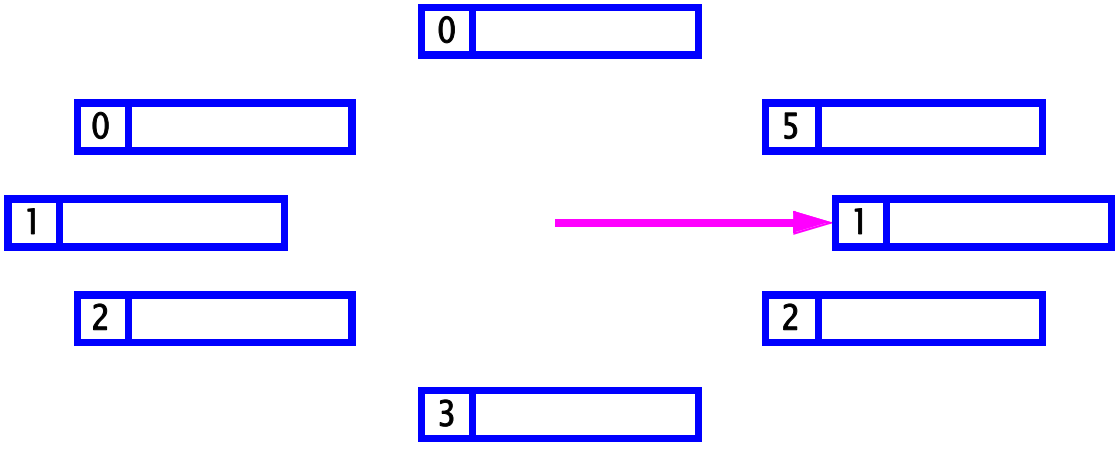
- Erweiterung von FIFO
- Referenzbit pro Seite, das bei Zugriff gesetzt wird
- Ersetzung erfolgt nur bei zurückgesetztem Bit (sonst erfolgt Zurücksetzen des Bits)



- annähernde Berücksichtigung des letzten Referenzierungszeitpunktes

GCLOCK (Generalized CLOCK)

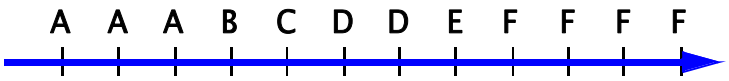
- pro Seite wird Referenzzähler geführt (statt Bit)
- Ersetzung nur von Seiten mit Zählerwert 0 (sonst erfolgt Dekrementierung des Zählers und Betrachtung der nächsten Seite)



- Verfahrensparameter:
  - Initialwerte für Referenzzähler
  - Wahl des Dekrementes
  - Zählerinkrementierung bei erneuter Referenz
  - Vergabe von seitentyp- oder seitenspezifischen Gewichten

Least-Reference-Density (LRD)

- Referenzdichte: Referenzhäufigkeit während eines bestimmten Referenzintervalls
- LRD Variante 1: Referenzintervall entspricht Alter einer Seite
- Berechnung der Referenzdichte:
  - Globaler Zähler GZ: Gesamtanzahl aller Referenzen
  - Einlagerungszeitpunkt EZ: GZ-Wert bei Einlesen der Seite
  - Referenzzähler RZ
  - ***Referenzdichte*** $RD(j) = \frac{RZ(j)}{GZ - EZ(j)}$



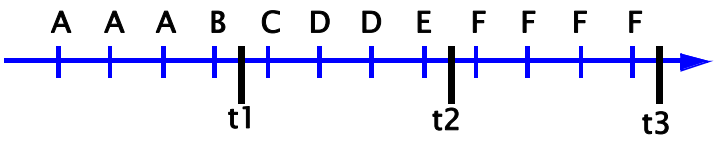
	RZ	EZ	RD
A			
B			
C			
D			
E			
F			

IDBS

Least-Reference-Density (2)

- LRD Variante 2: konstante Intervallgröße
- periodisches Reduzieren der Referenzzähler, um Gewicht früher Referenzen zu reduzieren
  - Reduzierung von RZ durch Division oder Subtraktion:

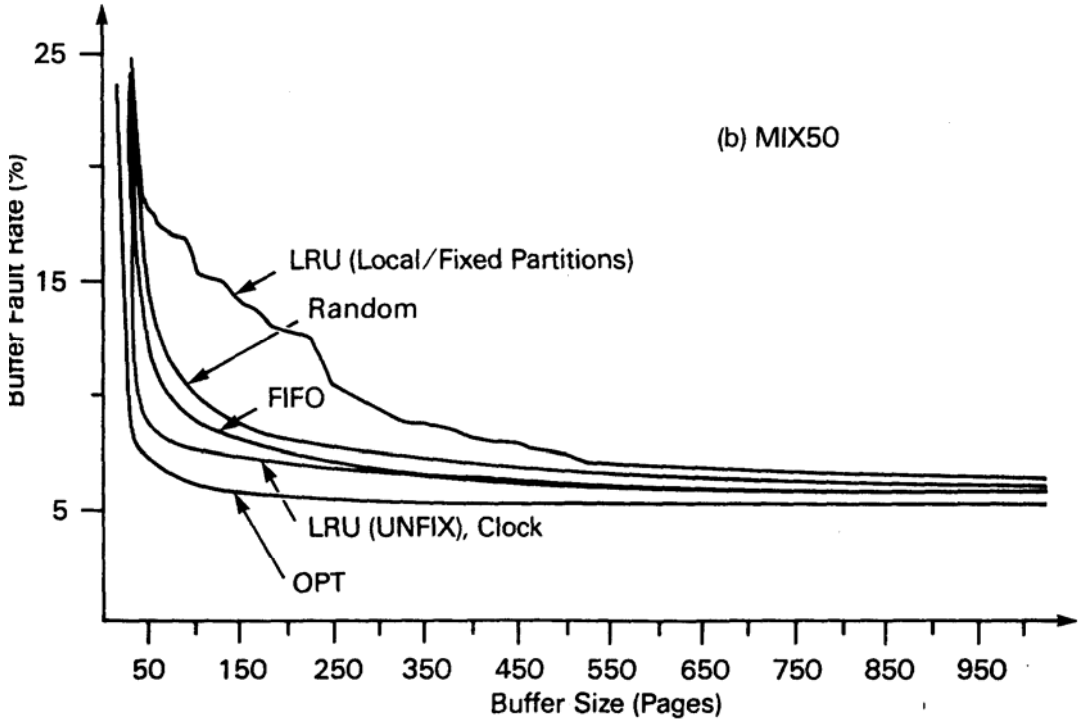
$$RZ(i) = \frac{RZ(i)}{K1} \quad (K1 > 1) \text{ oder } RZ(i) = \begin{cases} RZ(i) - K2 & \text{falls } RZ(i) - K2 \geq K3 \\ K3 & \text{sonst} \end{cases} \quad (K2 > 0, K3 \geq 0)$$



	t1	t2	t3
RZ(A)			
RZ(B)			
RZ(C)			
RZ(D)			
RZ(E)			

IDBS

Simulationsergebnisse

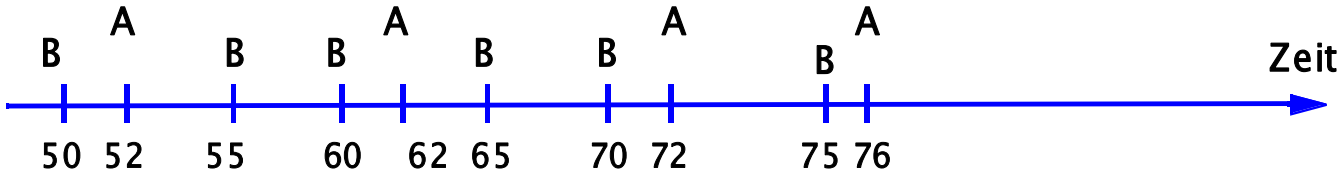


Probleme von LRU

- LRU ungeeignet für sequentielle Zugriffsmuster (z.B. Relationen-Scan)
  - abgearbeitete Seiten werden für dieselbe Query/Transaktion nicht mehr benötigt
  - sofortige Ersetzung sinnvoll (-> *Most Recently Used, MRU*)
- LRU nutzt kein Wissen über spezielle Referenzfolgen, z.B. zyklisches Referenzieren einer Menge von Seiten
  - zyklisches Referenzieren von S Seiten mit S > #Rahmen → internes Thrashing
  - zyklisches Referenzieren von S Seiten mit S < #Rahmen und Interferenz durch andere Transaktionen mit schnellerer Anforderung (stealing) → externes Thrashing
- LRU berücksichtigt nicht unterschiedliche Referenzhäufigkeiten, z.B. zwischen Index- und Datenseiten
- LRU berücksichtigt nicht Charakteristika unterschiedlicher Anfrage-/ Transaktionstypen
  - Bsp.: Transaktionen mit hoher Referenzlokalität können durch gleichzeitige sequentielle Scans mit schneller Seitenanforderung stark benachteiligt werden
- Alternativen
  - Ausnutzen von Kontextwissen des Query-Optimierers ("hints" an Pufferverwaltung)
  - LRU-Erweiterungen bzgl. Prioritäten, Referenzhäufigkeiten etc.

LRU-K

- Berücksichtigung der K letzten Referenzzeitpunkte einer Seite
  - erlaubt Approximation der Referenzhäufigkeit durch Bestimmung des mittleren Zeitabstands zwischen Referenzen einer Seite
  - Beschränkung auf die K letzten Referenzen ist einfache Methode, Information aktuell zu erhalten (keine zusätzlichen Tuning-Parameter wie bei LRD V2)
- Beispiel (K=4)

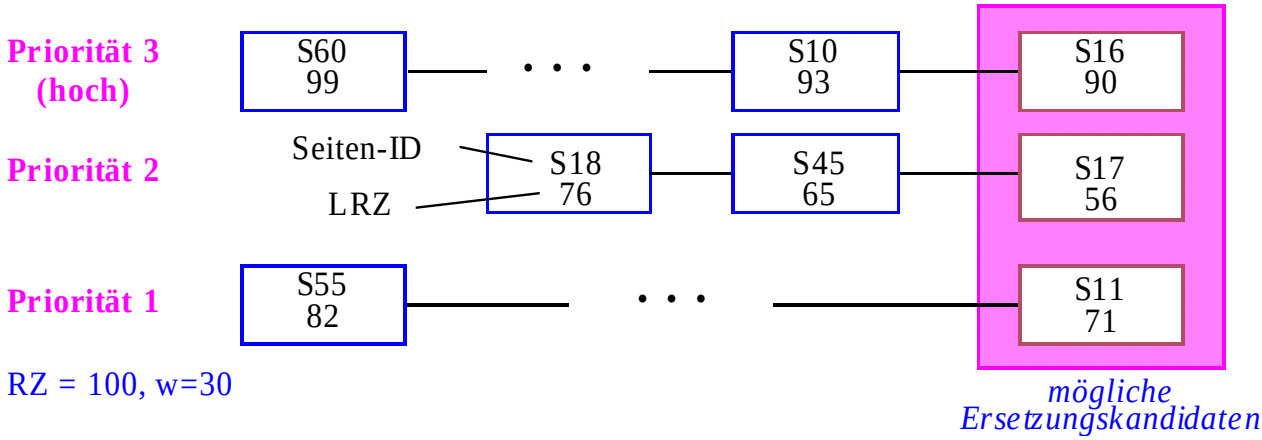


- zur Ersetzung genügt es, K-letzten Referenzierungszeitpunkt zu berücksichtigen!
- LRU-2 (d.h. K=2) stellt i.a. beste Lösung dar
  - ähnlich gute Ergebnisse wie für K > 2, jedoch einfachere Realisierung
  - bessere Reagibilität auf Referenzschwankungen als für größere K

IDBS

Prioritätsgesteuerte Seitenersetzung

- Bevorzugung bestimmter Transaktionstypen/DB-Partitionen
  - z.B. um Benachteiligungen durch sehr lange Queries oder sequentielle Zugriffe zu vermeiden
    => Berücksichtigung von Prioritäten bei der Pufferverwaltung
- Verfahren PRIORITY-LRU:
  - pro Prioritätsstufe eigene dynamische Pufferpartition; LRU-Kette pro Partition
  - Priorität einer Seite bestimmt durch DB-Partition bzw. durch (maximale) Priorität referenzierender Transaktionen
  - ersetzt wird Seite aus der Partition mit der geringsten Priorität. Ausnahme: die w zuletzt referenzierten Seiten sollen (unabhängig von ihrer Priorität) nicht ersetzt werden
  - Kompromiss zwischen Prioritäts- und absolutem LRU-Kriterium möglich

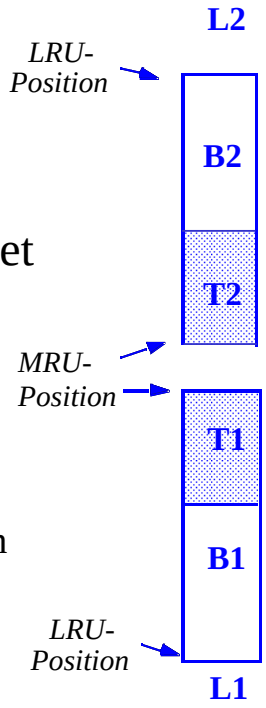


IDBS

Adaptives LRU*

- Verwendung von 2 LRU-Ketten
  - L1: Seiten die nur 1-mal referenziert wurden (v.a. für sequentielle Zugriffe)
  - L2: Seiten, die wenigstens 2-mal referenziert wurden
- für Cache-Größe c werden 2c Seiten in L1 und L2 verwaltet
  - jede Liste verwaltet in einem Top-Bereich (T1 bzw. T2) die gepufferten Seiten sowie
  - in einem Bottom-Bereich (B1 bzw. B2), die nicht mehr gepufferten Seiten

    |T1|+|T2|<=c ; |L1|+|L2| = |T1|+|T2|+|B1|+|B2| <=2c; |L1|<=c
  - falls angeforderte Seite in L1 oder L2 wird sie an MRU-Position von L2 gebracht, ansonsten an MRU-Position von L1
- Parameter p = |T1| bestimmt relatives Verhältnis zwischen L1- und L2-Seiten im Puffer
- dynamische Anpassung von p gemäß aktueller Verteilung von sequentiellen Zugriffen und Zugriffen mit temporaler Lokalität



* N. Megiddo, D.S. Modha: *Outperforming LRU with an Adaptive Replacement Cache Algorithm*. IEEE Computer, April 2004

Adaptives LRU (2)

- Ersetzung für Referenz auf Seite x

  1: Hit in T1 oder T2: Bringe x an MRU-Position von T2

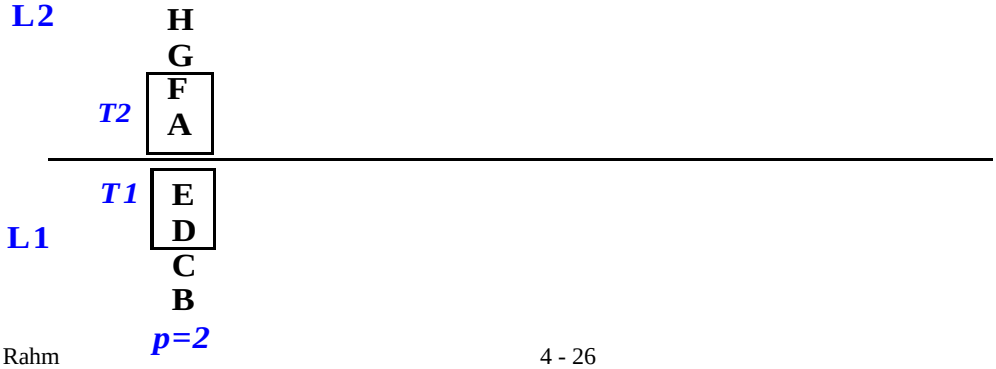
  2:"Hit" in B1: p := min (p+k1,c) mit k1 = 1 falls |B1| >=| B2|, sonst k1=|B2|/|B1|; Erhöhe p (|T1|)
  REPLACE(x,p); lese x ein und bringe x an MRU-Position von T2

  3: "Hit" in B2: p:=max(p-k2,0) mit k2=1 falls |B2|>=| B1|, sonst k2=|B1|/|B2|;
  Reduziere p (erhöhe |T2|)
  REPLACE(x,p); lese x ein und bringe x an MRU-Position von T2

  4: Miss: REPLACE (x,p); lese x ein und bringe x an MRU-Position von T1

  REPLACE (x,p): IF |T1| > 0 and (p<|T1| OR (p=|T1| and (x in B2))
  Ersetze LRU-Seite von T1 (-> Wechsel an MRU-Position von B1)
  ELSE Ersetze LRU-Seite von T2 (-> Wechsel an MRU-Position von B2)

Beispiel ***(c=4)*** **G C A K F** *Referenzen*



Adaptives LRU: Beobachtungen

- Adaptivität über erweiterte Listen
- Heuristik: investiere in erfolgreiche Liste (vergrößere T1 bzw. T2 für Treffer in B1 bzw. B2
- rein sequentielle Zugriffe füllen L1; fehlende Treffer in B1 vergrößern T2

Zusammenfassung

- Referenzmuster in DBS sind Mischformen
  - sequentielle, zyklische, wahlfreie Zugriffe
  - Lokalität innerhalb und zwischen Transaktionen
  - Seiten mit unterschiedlich hoher Referenzdichte
- Verwaltungsaufgaben:
  - Suche im Puffer: durch Hash-Verfahren
  - **Speicherzuteilung**: globale (Pufferrahmen für alle Transaktionen) vs. lokale Strategien (Sonderbehandlung bestimmter Anfragen bzw. DB-Bereiche)
  - **Behandlung geänderter Seiten**: NOFORCE, asynchrones Ausschreiben
  - Seitenersetzung: erfordert Vorhandensein von Lokalität (sonst Verhalten wie ~ RANDOM)
- Seitenersetzungsverfahren
  - "zu genaue" Verfahren sind schwierig einzustellen (⇒ instabil)
  - Nutzung mehrerer Kriterien möglich: Alter, letzte Referenz, Referenzhäufigkeit
  - LRU ist guter Default-Ansatz
- Erweiterungen von LRU
  - LRU-2 wählt Ersetzungskandidaten aufgrund des vorletzten Referenzzeitpunktes aus
  - Berücksichtigung von Prioritäten
  - Adaptives LRU