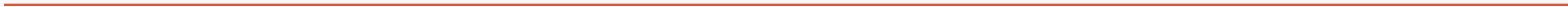


Upravljanje radnom memorijom

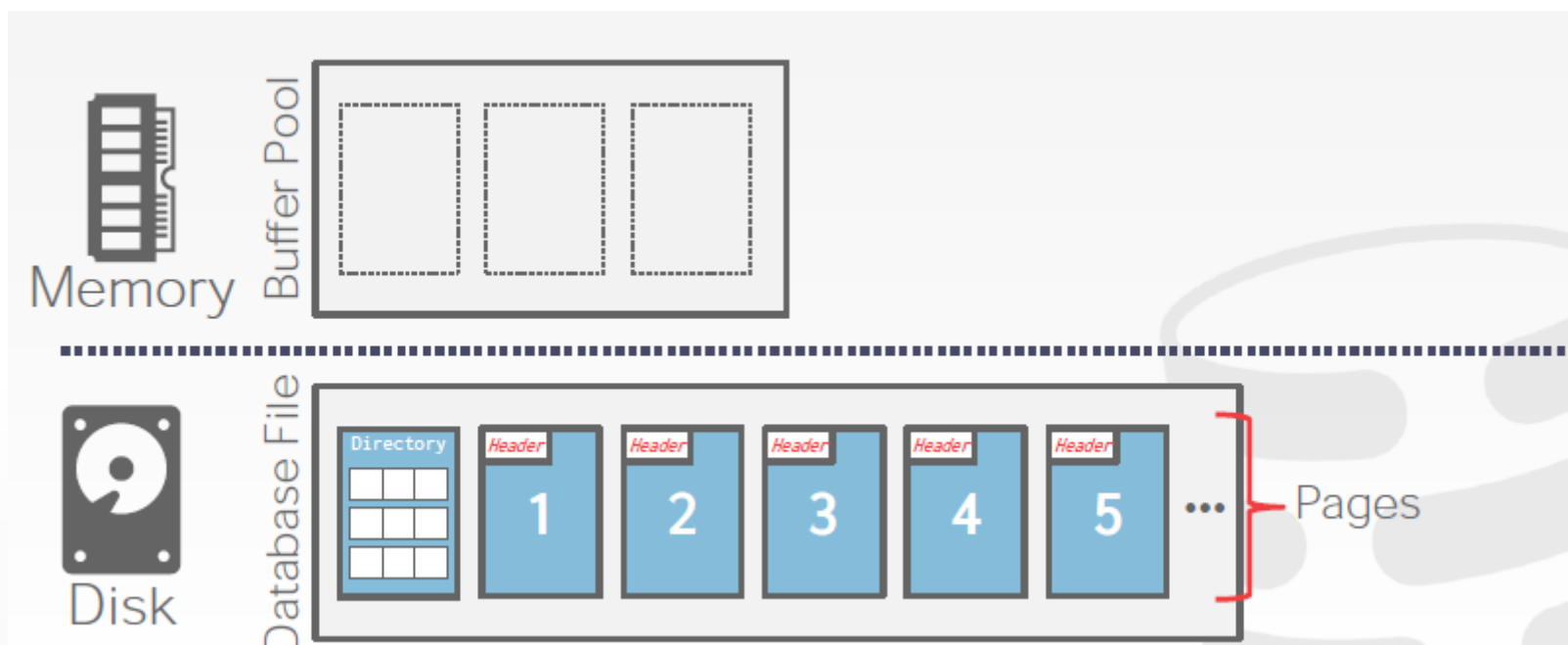
Buffer Management

Upravljanje memorijom

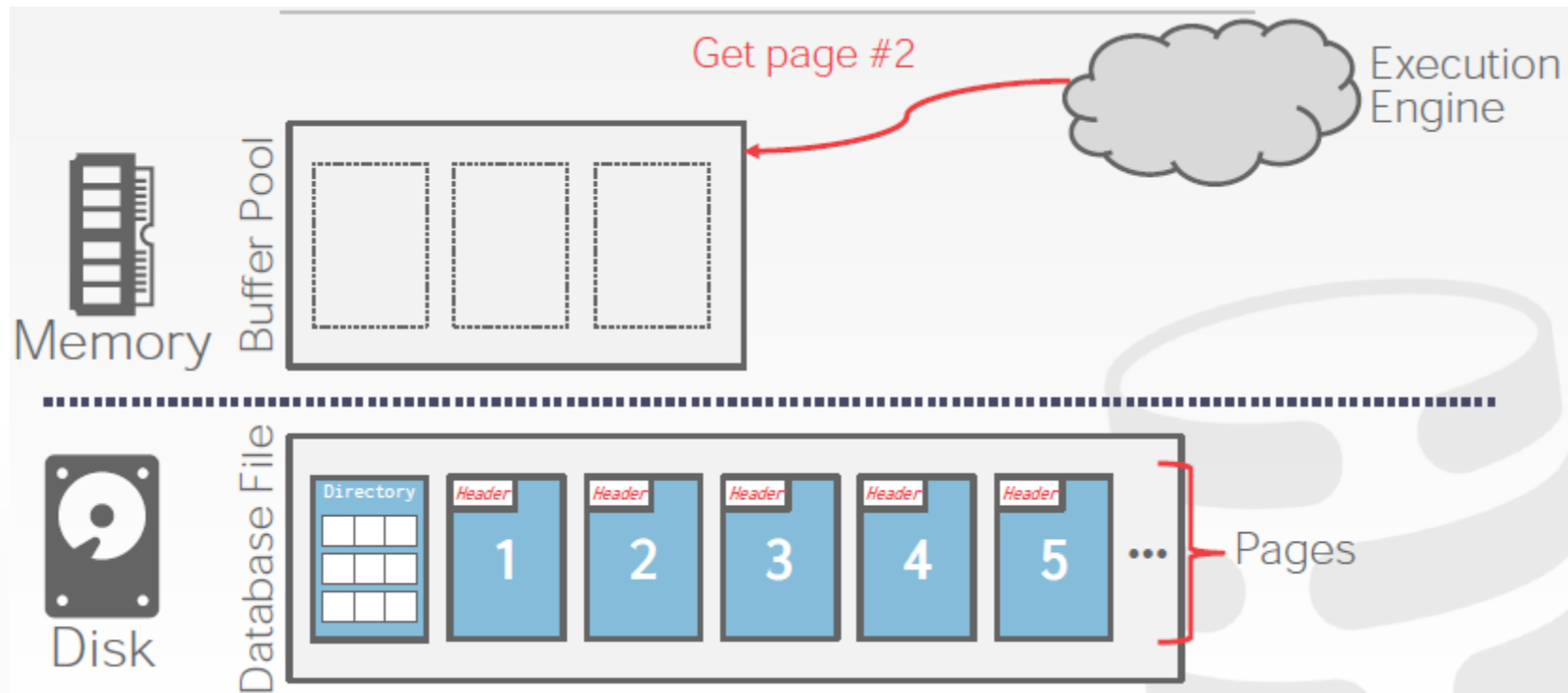
- Cilj -> minimizacija čekanja na učitavanje podataka sa diska.
 - Kada čitati, a kada pisati na disk?



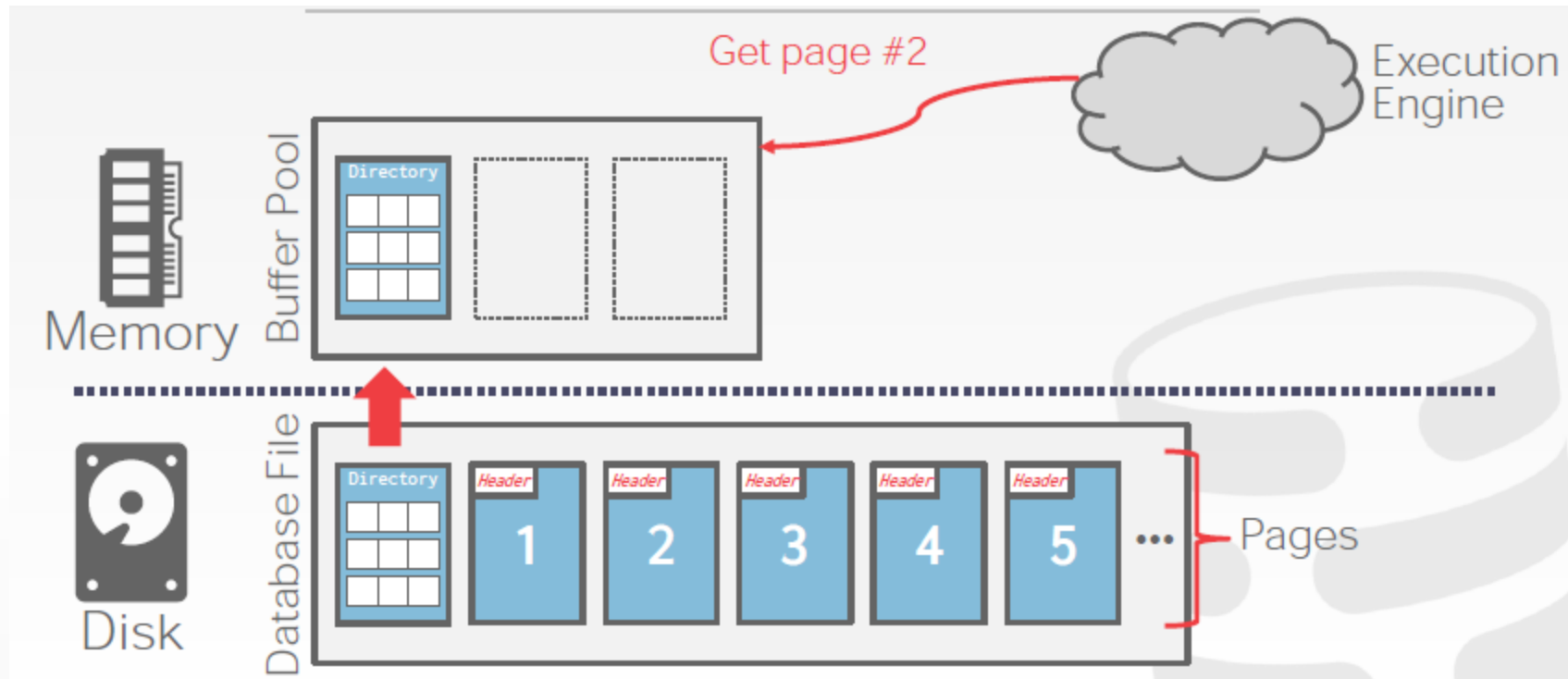
Bafer menadžer



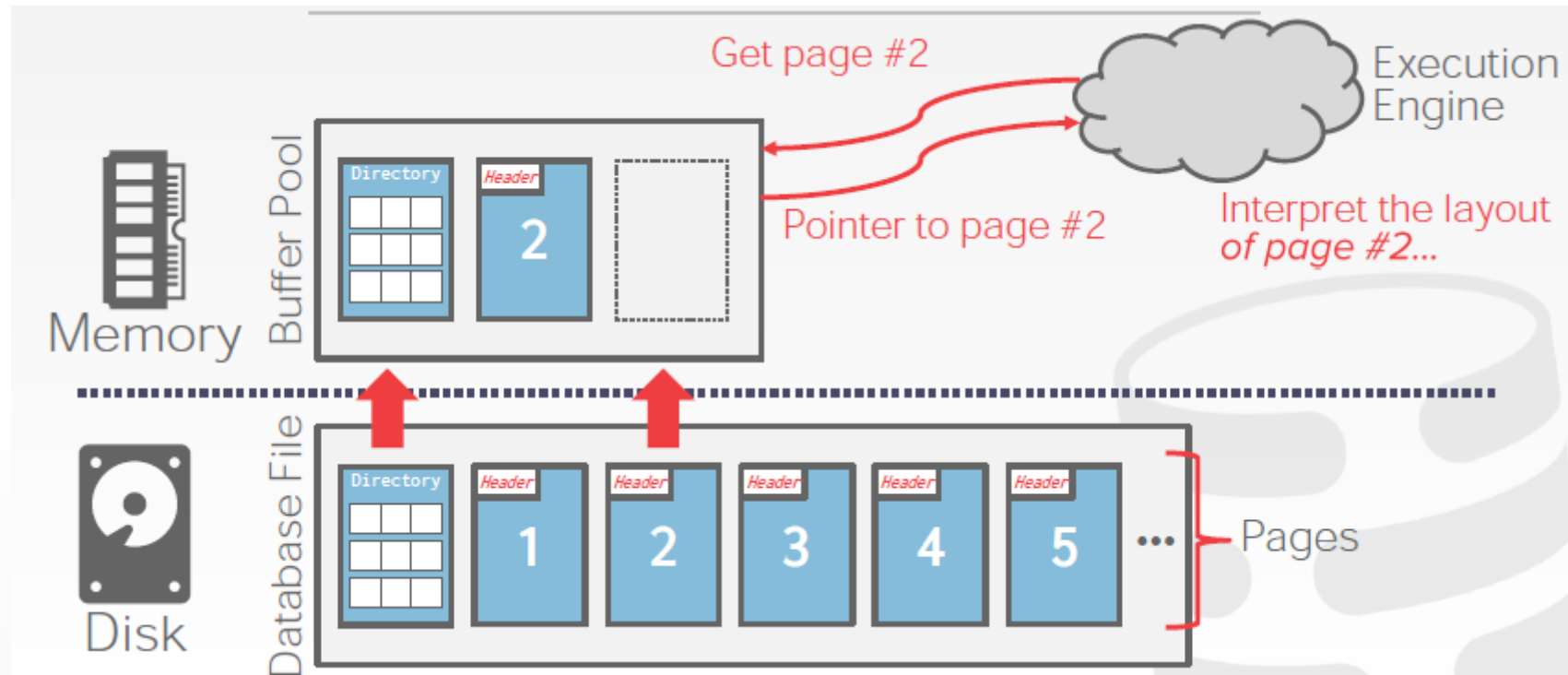
Bafer menadžer



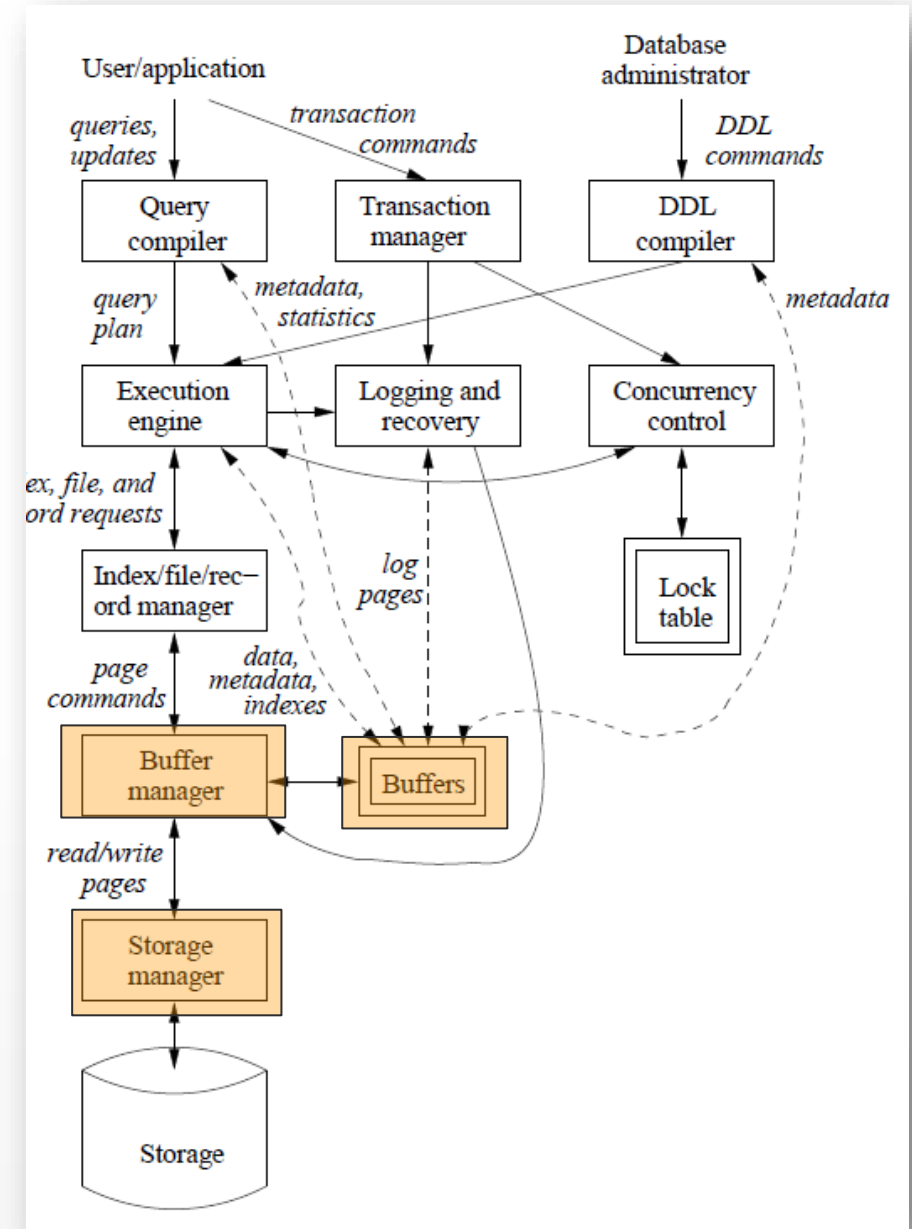
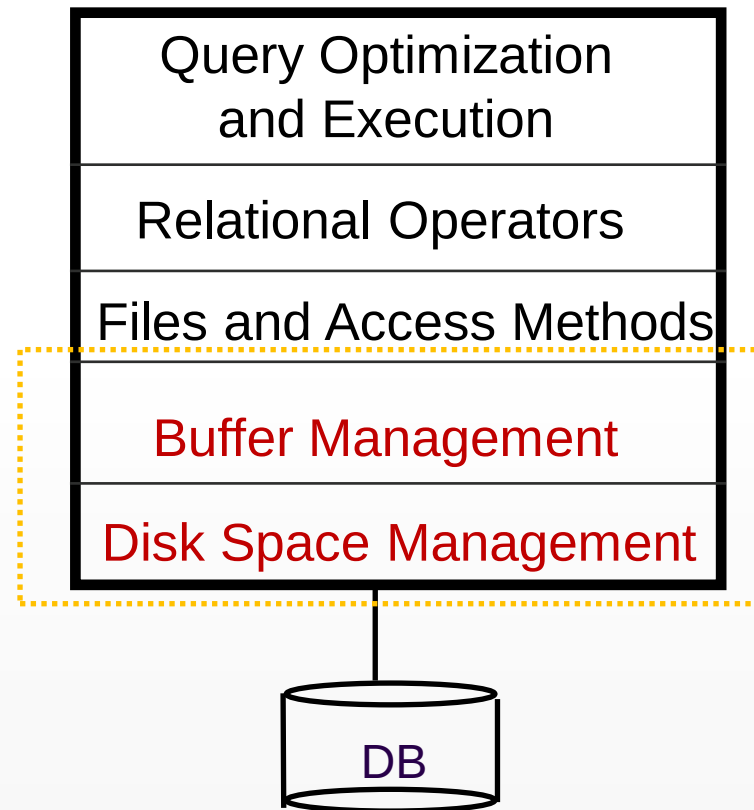
Bafer menadžer



Bafer menadžer

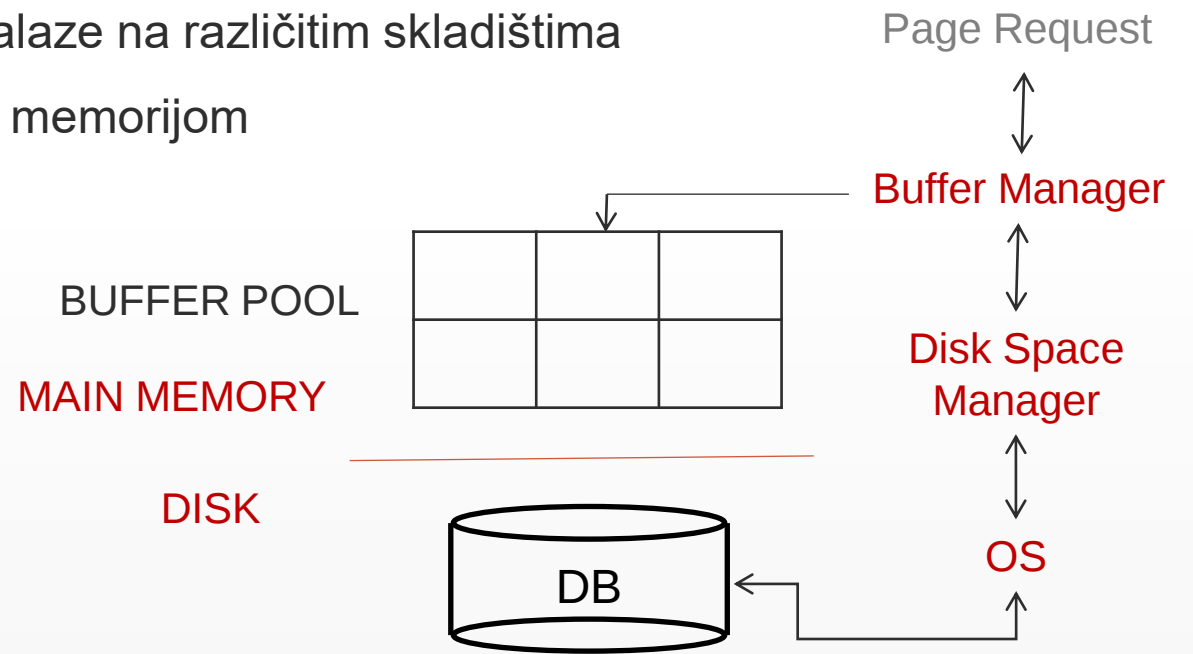


DBMS kontekst

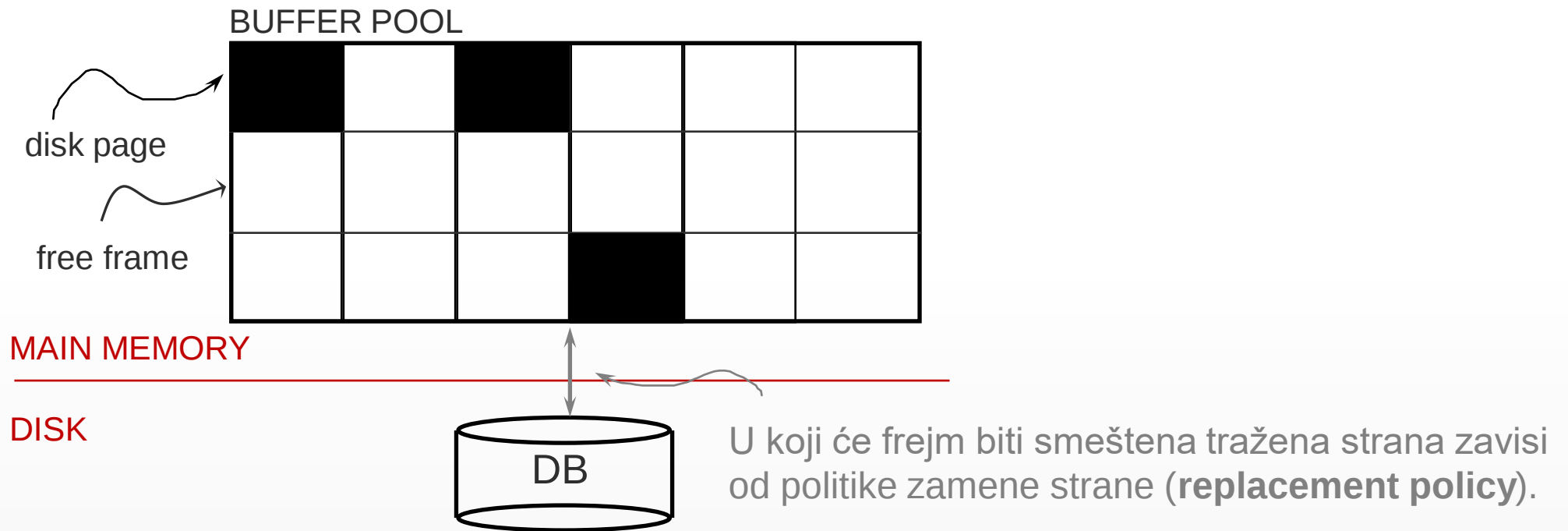


Bafer menadžer

- Sloj iznad disk menadžera
 - Zadužen za dobavljanje podataka u radnu memoriju
 - Sakriva činjenicu o tome da se podaci nalaze na različitim skladištima
- Bafer menadžer upravlja dostupnom radnom memorijom
- Memoriju uređuje kao **kolekciju strana**, koja se naziva **bafer pul (buffer pool)**.



Bafer menadžer



Frame – slot/mesto za stranu u baferu

Bafer menadžer

- Viši slojevi koji zahtevaju stranu, moraju da obaveste bafer menadžer:
 - da žele da **oslobode** stranu koja im više nije potrebna
 - da su **izmenili** zahtevanu stranu, da bi nakon toga bafer menadžer obezbedio beleženje izmenjene kopije na disk
- Bafer menadžer održava **Page Table**, tabelu koja sadrži:

<frame#, pageid, pin_count, dirty_bit>

pin_count - koliko puta je strana koja se nalazi u frejmu zahtevana, a nije oslobođena, tj. broj trenutnih korisnika

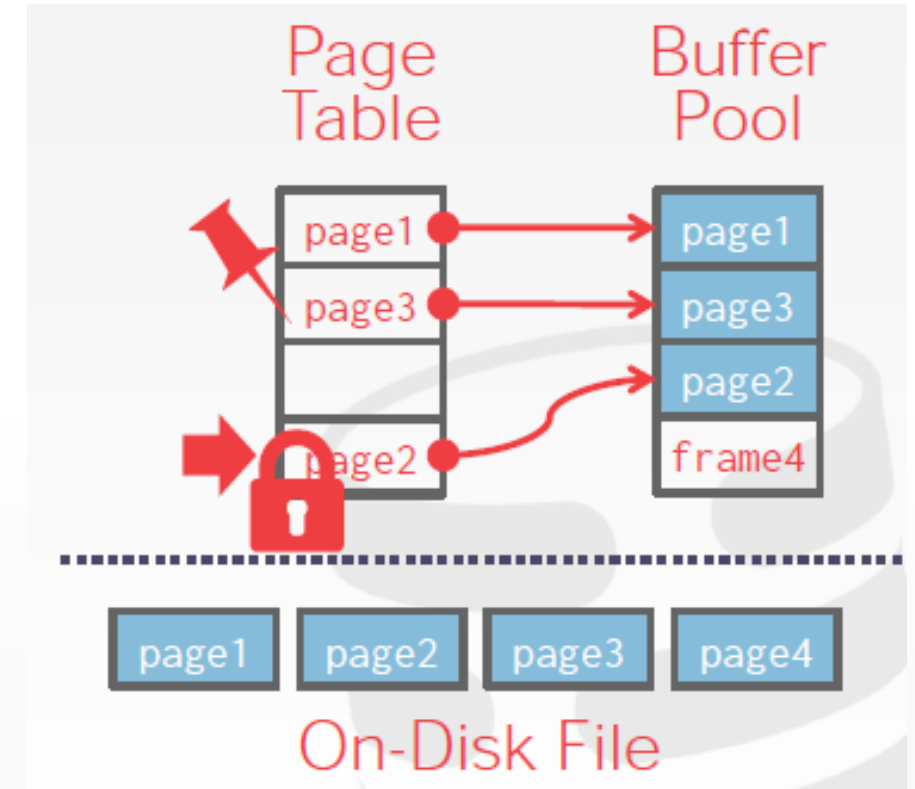
dirty_bit - boolean promenljiva koja je postavljena na true ukoliko je strana koja se nalazi u frejmu menjana od trenutka kada je dopremljena u pul.

Page table vs. Page Directory

- Page Directory
 - Mapira PID na lokacije stranica u db fajlovima.
 - Mora da bude čuvana kao i sve ostale stranice.
 - Page Table
 - Mapira PID na frejmove, tj. kopije stranica (učitane iz DB fajla) koje se nalaze u nekom frejmu
 - Unutrašnja struktura koja se ne čuna na disku.
-

Bafer pul tabela informacija

- Kada je bafer menadžeru upućen zahtev za nekom stranom, tada:
 - Ako je strana već u pulu tada njen brojač referenciranja uvećava za jedan - `pin_count++`
 - Ako strana nije u pulu tada se
 - Odabira frejm čiji će sadržaj biti **zamenjen**
Samo strane sa `pin_count=0`
 - Ako je **dirty=true**, pisanje na disk
 - Postavljanje reze (Latch - Mutex) na slog u tabeli strana (Page Table) čime se rezerviše frejm u koji će biti upisana nova stranica
 - Učitavanje tražene strane u markirani frejm



Lock ≠ Latch

- Lock (brava) štiti logički elemente (tabele) baze od neregularnog uticaja od strane konkurentnih transakcija i važi tokom trajanja transakcije koja ga je postavila.
 - Latch štiti interne DBMS strukture od uticaja drugih niti i važi tokom trajanja operacije.
-

Politika alokacije

- Globalna politika: Donošenje odluka o alokaciji frejmova sa ciljem optimizacije rada celog sistema.
 - Lokalna politika: Alokacija frejmova za pojedinačne tredove/upite.
 - Većina Sistema koristi kombinaciju.
-

Optimizacija rada bafer pula

- Višestruki bafer pulovi
 - Pre-Fetching
 - Scan Sharing
 - Zaobilazanje bafer pula
-

Višestruki bafer pulovi

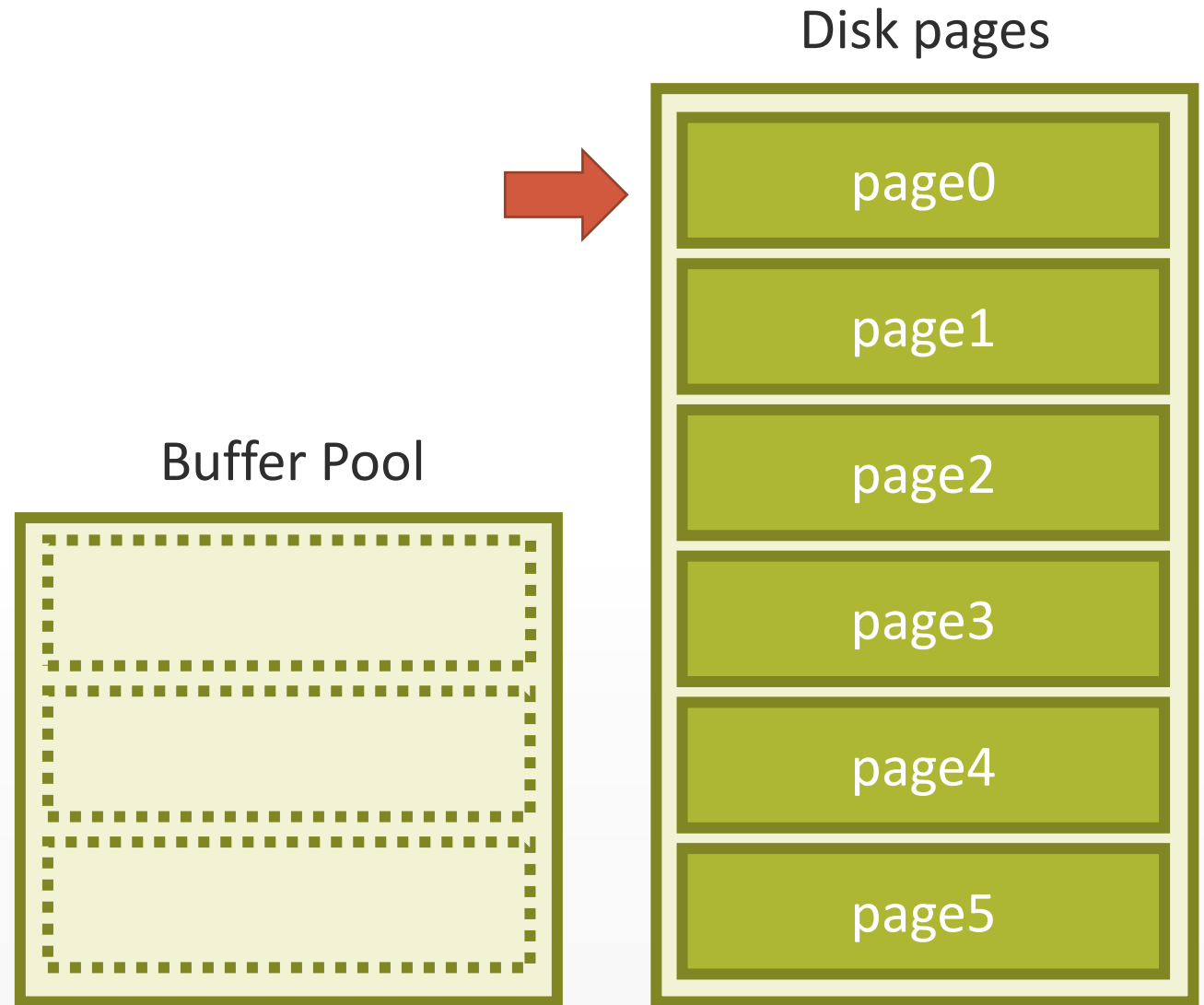
- Većina DBMS-ova ne koristi samo jedan bafer pul za podršku radu celog sistema/svih tredova.
 - Per-database buffer pool
 - Per-page type buffer pool
 - Per-table buffer pool
-

PRE-FETCHING

- Ukoliko se zahtevi mogu predvideti (sekventcijalna pretraga) tada se strane mogu učitavati i unapred – **pre-fetched**
 - Na osnovu plana izvršavanja upita DMBS-ovi (mogu da) vrše dobavljanje stranica pre nego što je za njima iskazana potreba.
 - Primeri:
 - Sekvencijalno čitanje
 - Sekvencijalna pretraga indeksa
-

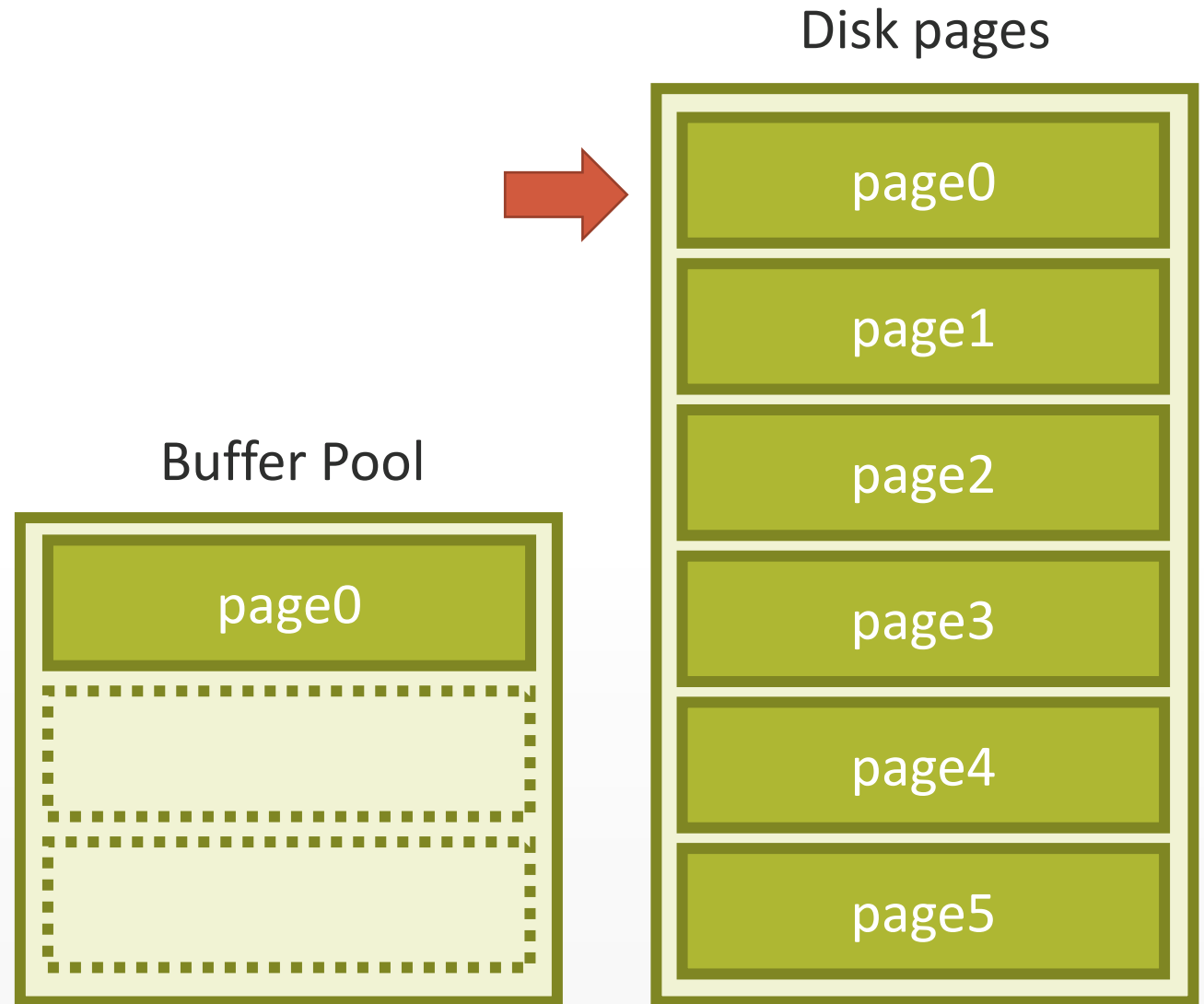
PRE-FETCHING

- Sekvencijalno čitanje



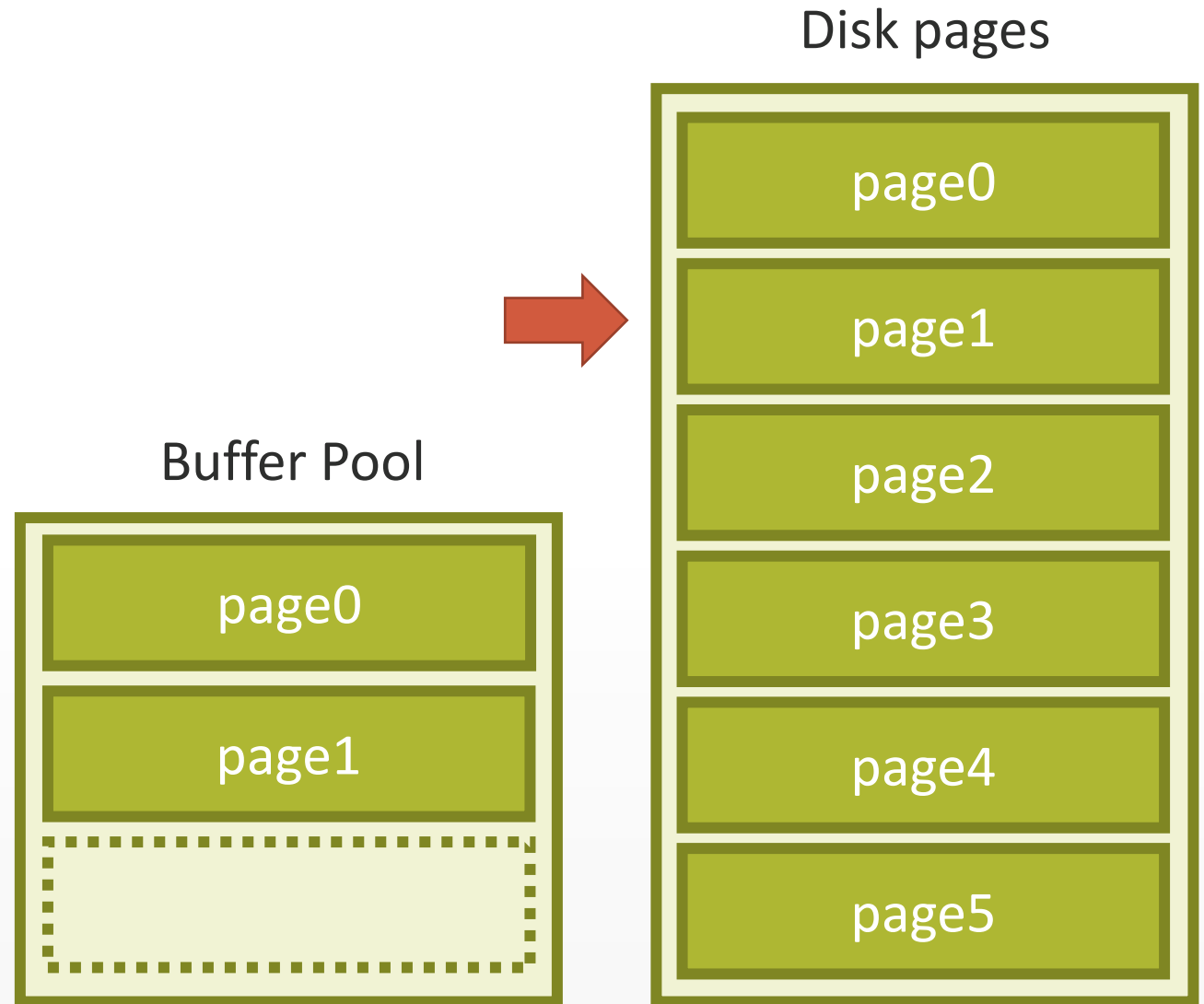
PRE-FETCHING

- Sekvencijalno čitanje



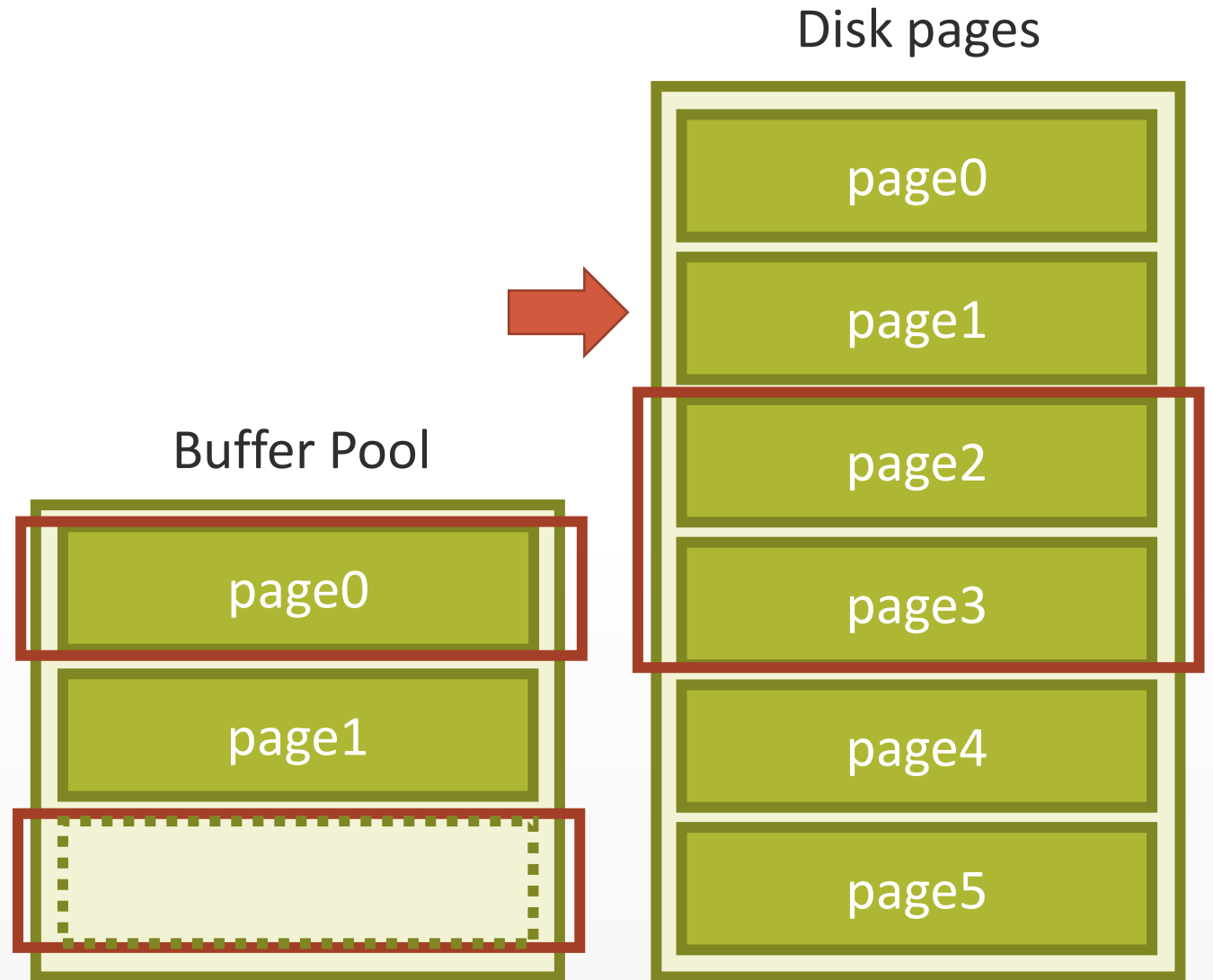
PRE-FETCHING

- Sekvencijalno čitanje



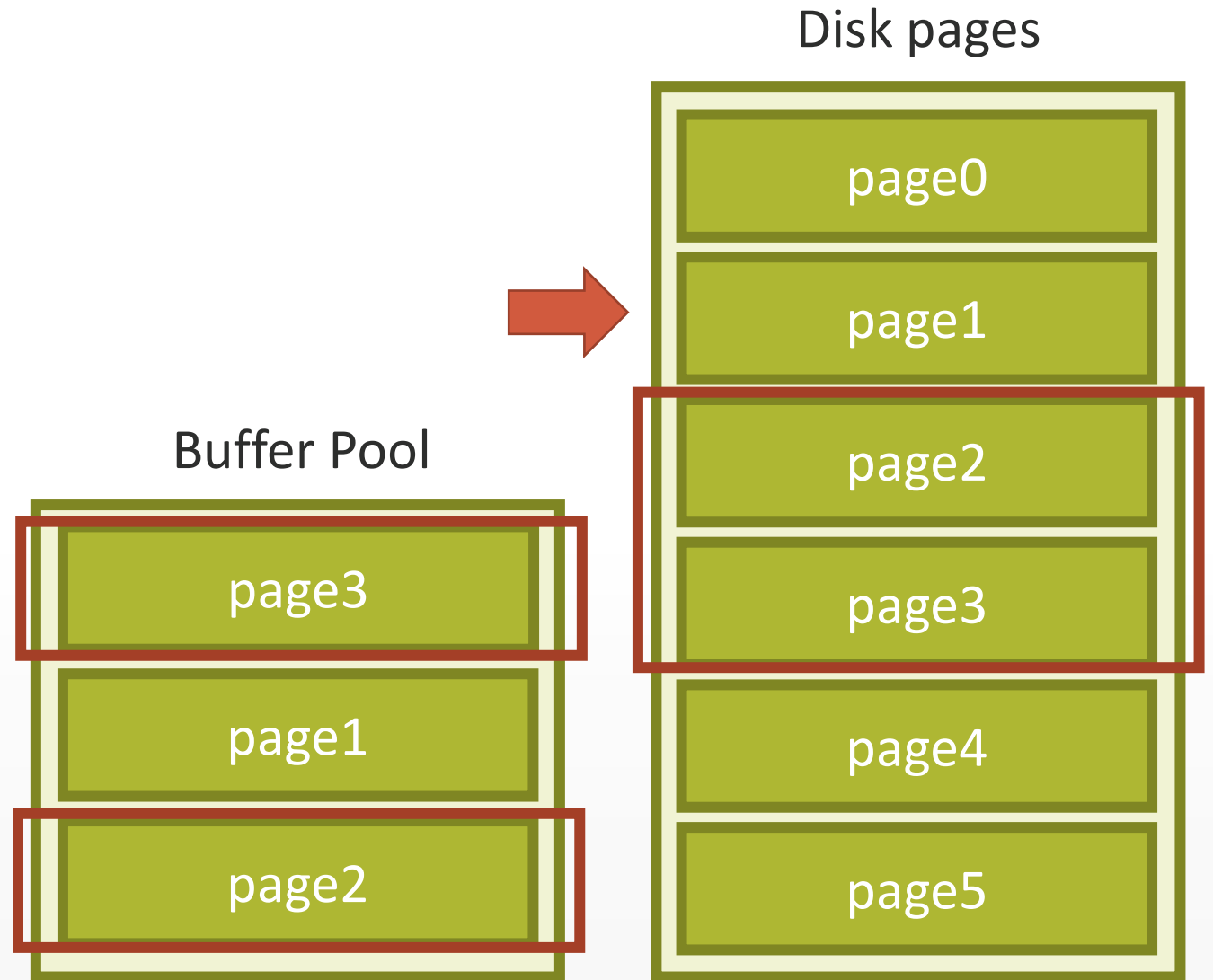
PRE-FETCHING

- Sekvencijalno čitanje



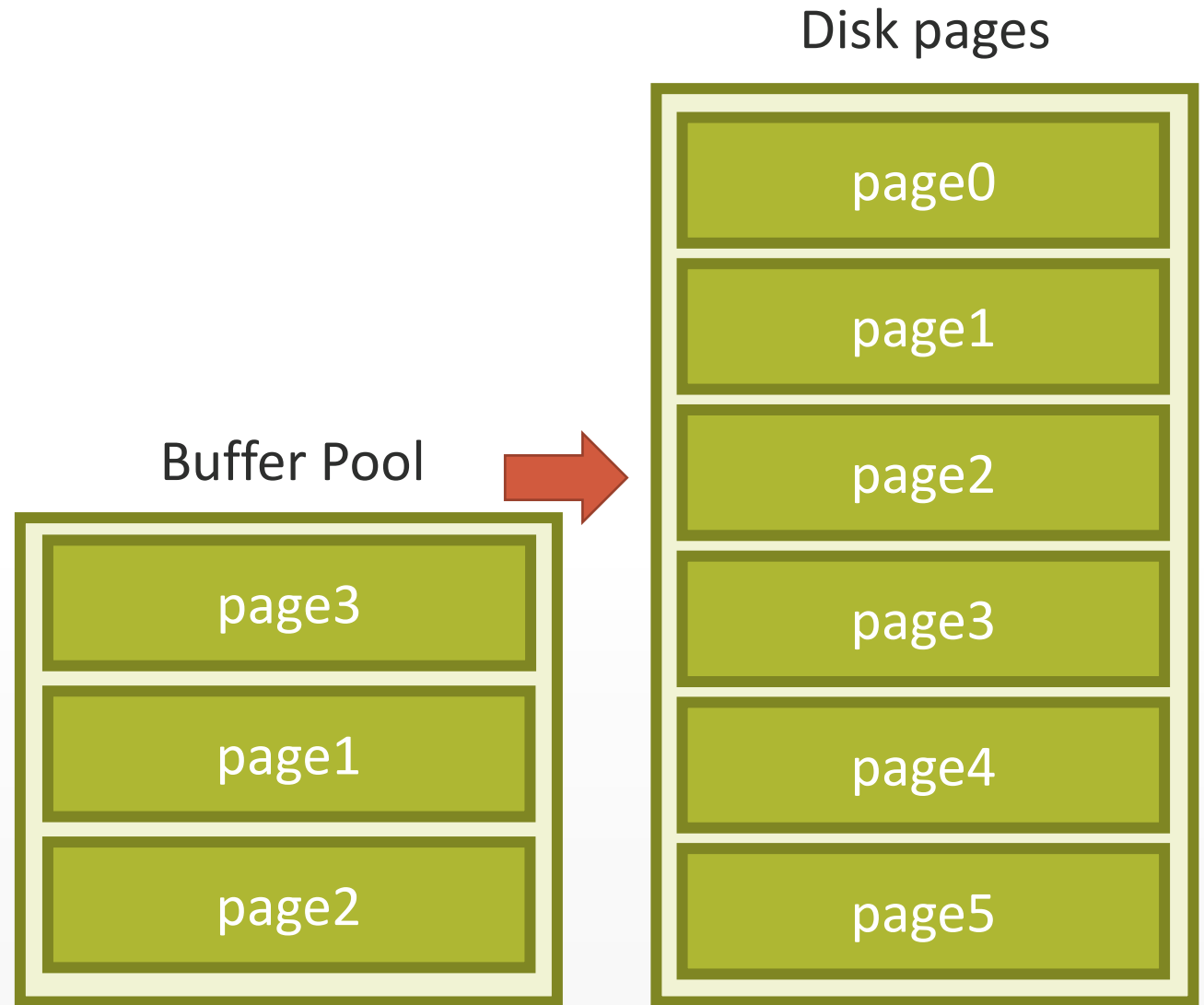
PRE-FETCHING

- Sekvencijalno čitanje

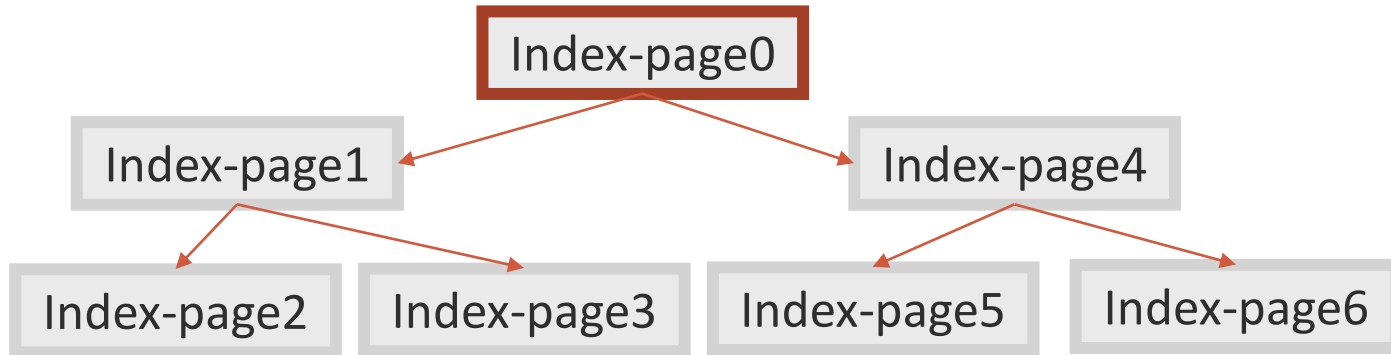


PRE-FETCHING

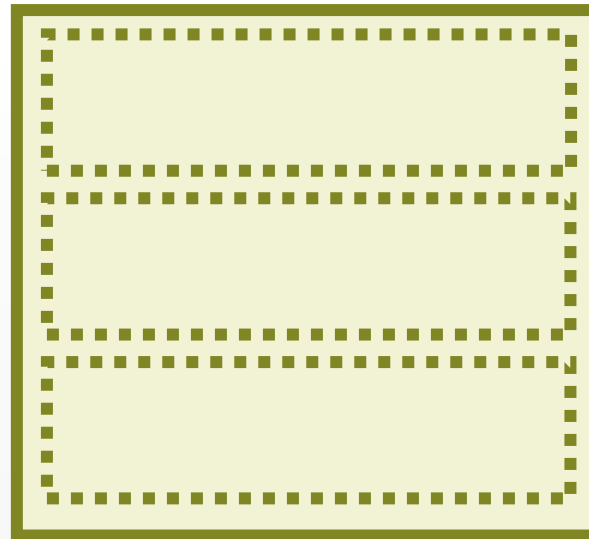
- Sekvencijalno čitanje



PRE-FETCHING



Buffer Pool

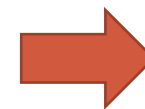
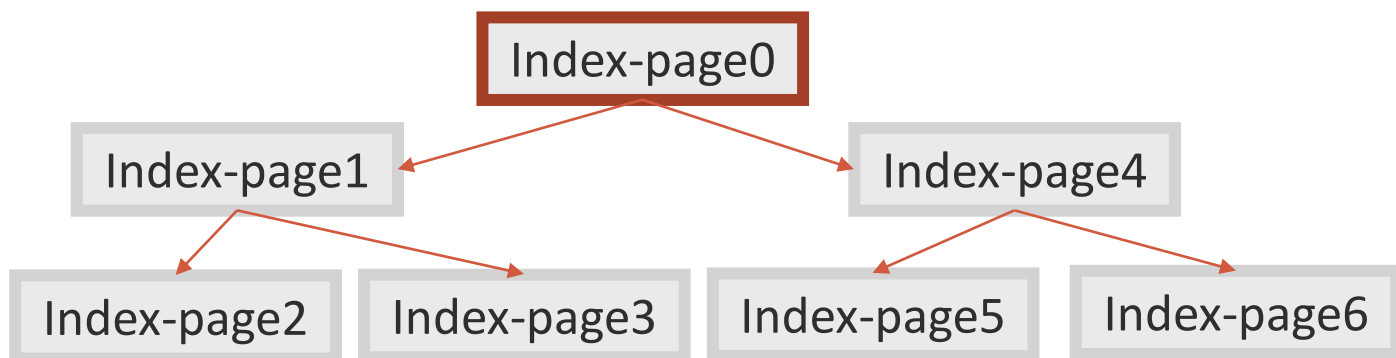


Disk pages



- Sekvencijalno čitanje indeksa

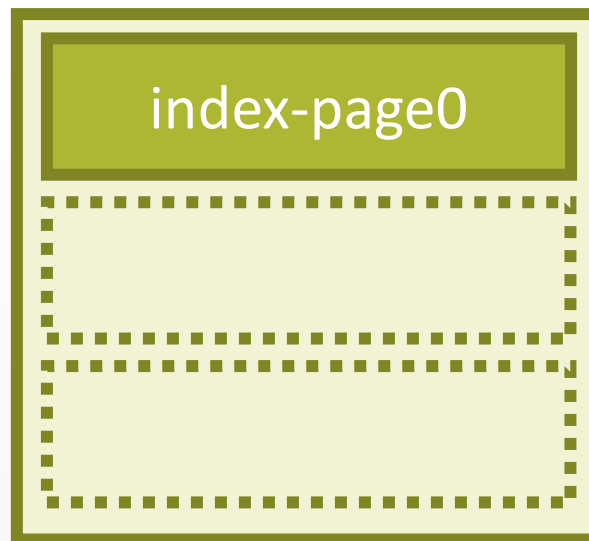
PRE-FETCHING



Disk pages

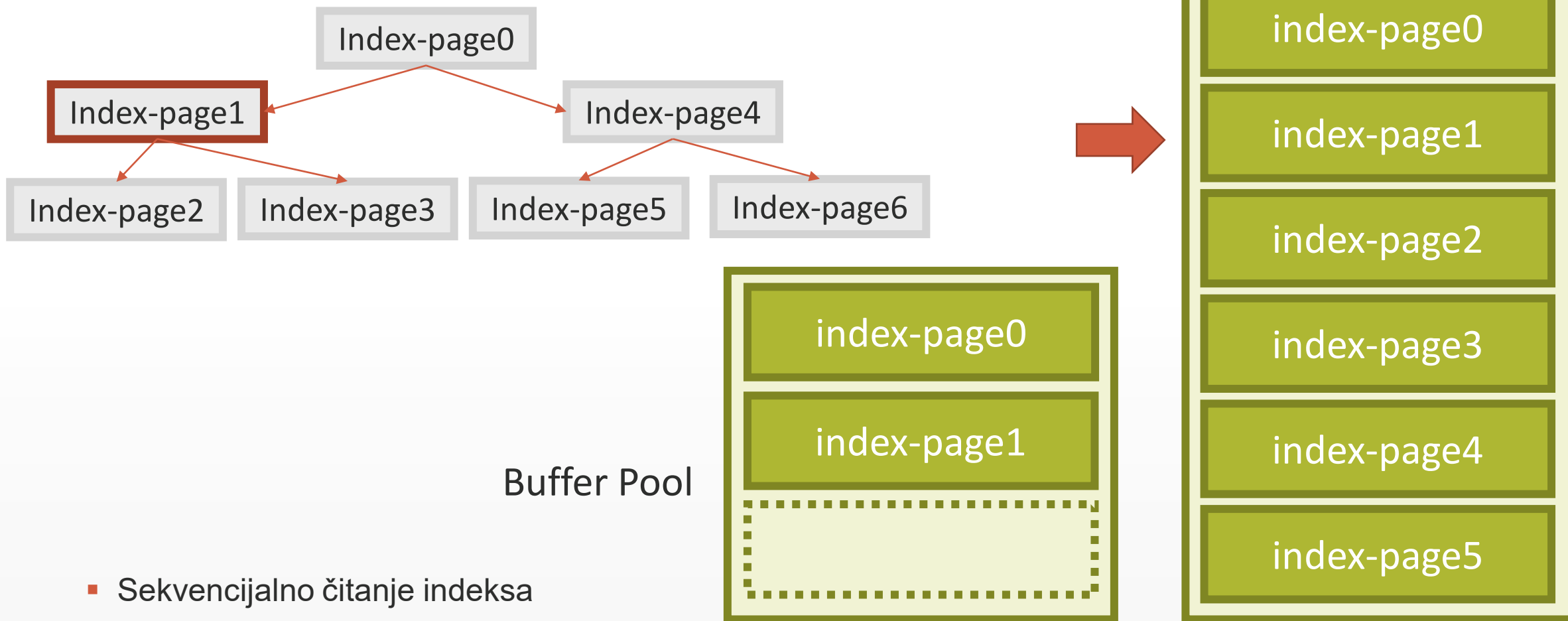


Buffer Pool

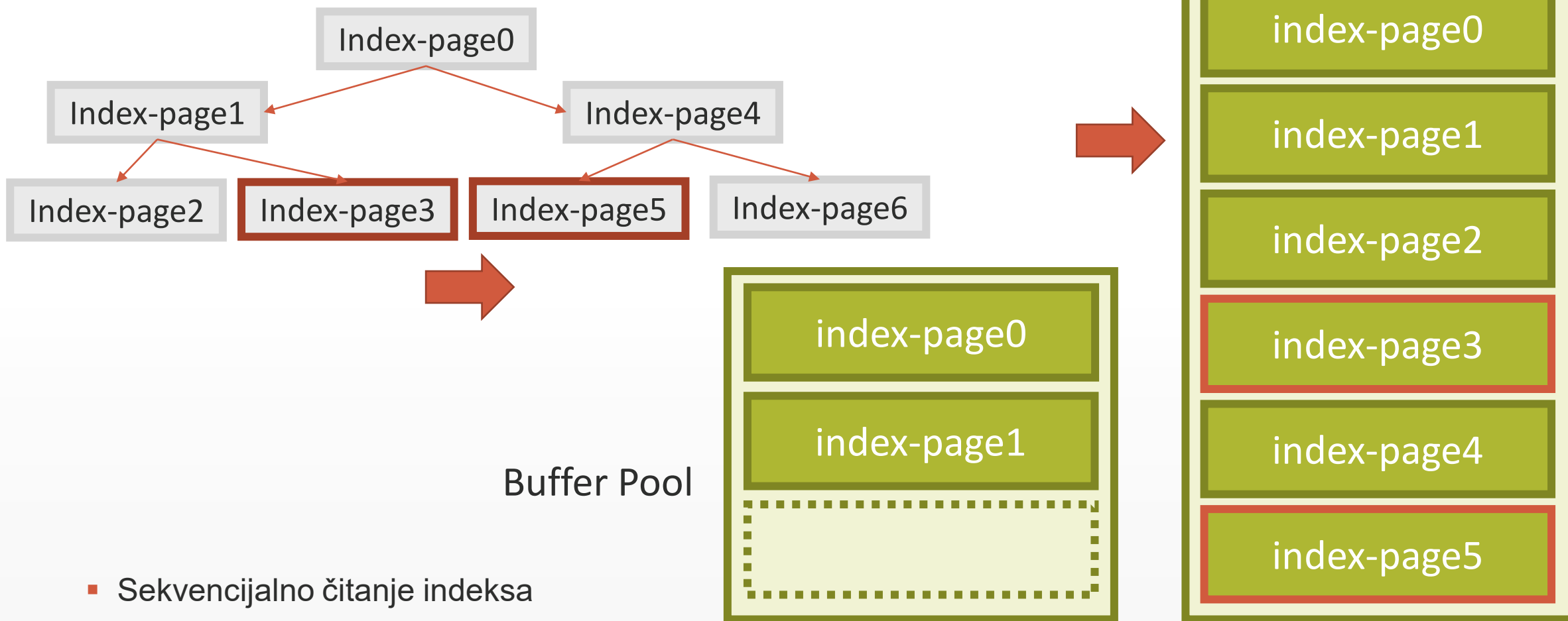


- Sekvencijalno čitanje indeksa

PRE-FETCHING



PRE-FETCHING



DELJENJE KURSORA

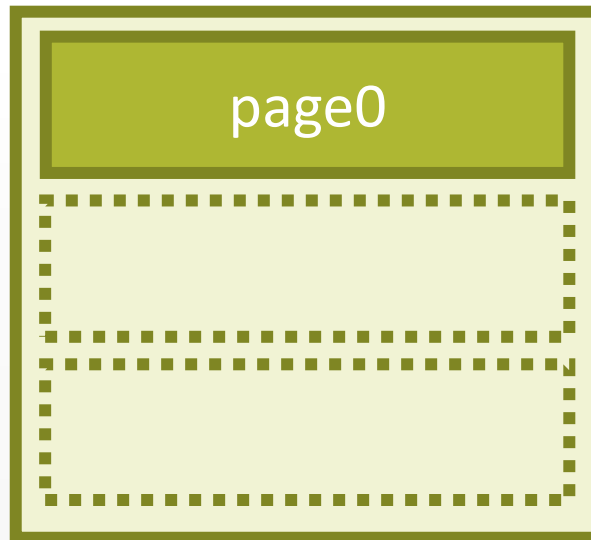
- DBMS može da dozvoli da više upita koristi jedan kursor za čitanje tabele ili međurezultata.
 - Ako se pokrene izvršavanje upita koji čita podatke koje već neko drugi čita, DBMS će preusmeriti kursor na postojeći.
 - DMS vodi evidenciju o tome gde se novi upis pridružio u čitanju.
 - MSSQL, IBM DB2
 - Oracle podržava deljenje kursora samo za identične upite.
-

DELJENJE ČITANJA

Q1

```
SELECT SUM(VAL) FROM A
```

Buffer Pool



Q1 →

Disk pages



DELJENJE ČITANJA

Q1

```
SELECT SUM(VAL) FROM A
```

Buffer Pool



Disk pages

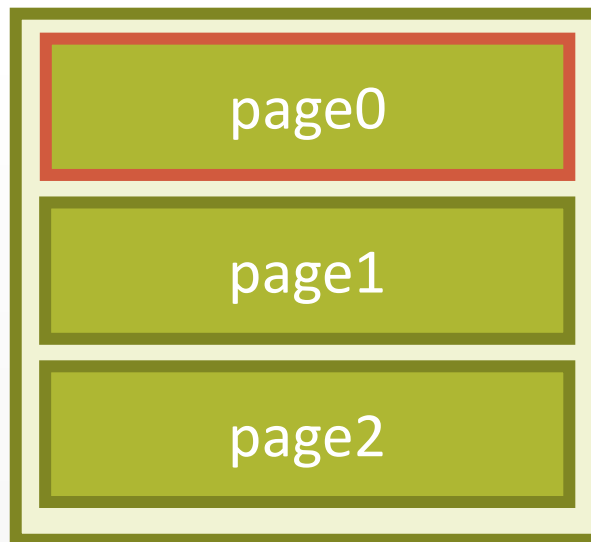


DELJENJE ČITANJA

Q1

```
SELECT SUM(VAL) FROM A
```

Buffer Pool



Disk pages

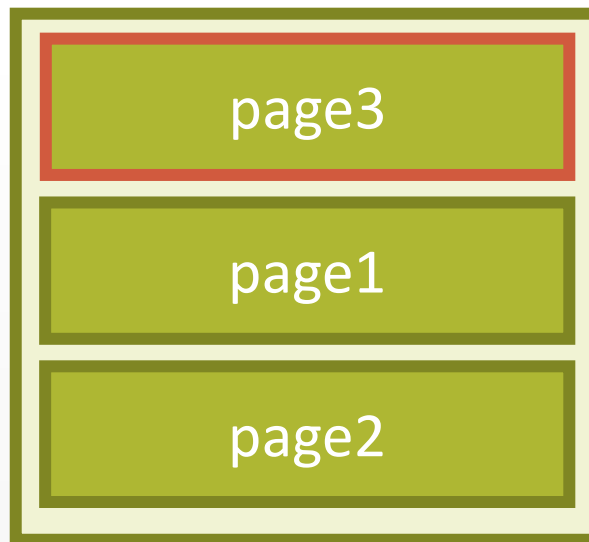


DELJENJE ČITANJA

Q1

```
SELECT SUM(VAL) FROM A
```

Buffer Pool



Disk pages



DELJENJE ČITANJA

Q1

SELECT SUM(VAL) FROM A

Q2

SELECT AVG(VAL) FROM A

Buffer Pool



Disk pages



DELJENJE ČITANJA

Q1

SELECT SUM(VAL) FROM A

Q2

SELECT AVG(VAL) FROM A

Buffer Pool



Disk pages



DELJENJE ČITANJA

Q1

SELECT SUM(VAL) FROM A

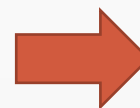
Q2

SELECT AVG(VAL) FROM A

Buffer Pool



Q1
Q2



Disk pages



DELJENJE ČITANJA

Q1

```
SELECT SUM(VAL) FROM A
```

Q2

```
SELECT AVG(VAL) FROM A
```

Buffer Pool



Q2 →

Disk pages



DELJENJE ČITANJA

Q1

```
SELECT SUM(VAL) FROM A
```

Q2

```
SELECT AVG(VAL) FROM A
```

Buffer Pool



Disk pages



Politika zamene

- Politikom zamene se definiše način na koji se određuje koji će slot u pulu biti popunjen novim sadržajem.
 - Ciljevi:
 - Korektnost – ne dozvoliti prepisivanje pinovane strane
 - Preciznost – dobra procena toga koja stranica verovatno neće biti ponovo tražena u narednom periodu
 - Brzo odlučivanje
 - Minimalni overhead koji nastaje uvođenjem meta podataka
 - Dobro razvijeni DBMSovi koriste sofisticirane politike zamene.
 - Vode statistike o korišćenosti strana i na osnovu njih procenjuju šta će biti korišćeno.
-

Politika zamene

- LRU – last recently used
 - Clock
 - LRU-k
-

LRU

- Pamti se vreme pristupa stranici u pulu. Kada je pul pun, a tražena stranica nije u pulu uklanja se stranica koja je najranije referencirana.

4 RAM frames	Req.	c	a	d	b	e	b	a	b	c	d
		c	c	c	c						
			a	a	a						
				d	d						
					b						
		F	F	F	F						

LRU

- Pamti se vreme pristupa stranici u pulu. Kada je pul pun, a tražena stranica nije u pulu uklanja se stranica koja je najranije referencirana.

c a d b
←—————

Req.	c	a	d	b	e	b	a	b	c	d
	c	c	c	c	e					
		a	a	a	a					
			d	d	d					
				b	b					
	F	F	F	F	F					

4 RAM frames

LRU

- Pamti se vreme pristupa stranici u pulu. Kada je pul pun, a tražena stranica nije u pulu uklanja se stranica koja je najranije referencirana.

4 RAM frames	Req.	c	a	d	b	e	b	a	b	c	d
		c	c	c	c	e	e	e	e		
			a	a	a	a	a	a	a		
				d	d	d	d	d	d		
					b	b	b	b	b		
		F	F	F	F	F					

LRU

- Pamti se vreme pristupa stranici u pulu. Kada je pul pun, a tražena stranica nije u pulu uklanja se stranica koja je najranije referencirana.

		d ← e a b									
Req.		c	a	d	b	e	b	a	b	c	d
4 RAM frames		c	c	c	c	e	e	e	e	e	
			a	a	a	a	a	a	a	a	
				d	d	d	d	d	d	c	
					b	b	b	b	b	b	
		F	F	F	F	F				F	

LRU

- Pamti se vreme pristupa stranici u pulu. Kada je pul pun, a tražena stranica nije u pulu uklanja se stranica koja je najranije referencirana.

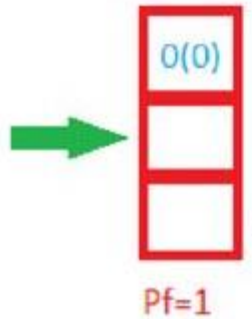
		e a b c ←									
4 RAM frames	Req.	c	a	d	b	e	b	a	b	c	d
		c	c	c	c	e	e	e	e	e	d
			a	a	a	a	a	a	a	a	a
				d	d	d	d	d	d	c	c
					b	b	b	b	b	b	b
		F	F	F	F	F				F	F

Clock (second chance)

- Ne pamte se vremena pristupa, već za svaku stranu po jedan reference bit.
 - Kad god se strani koja se nalazi u pulu pristupa, reference bit se postavlja na 1.
 - Informacije o stranicama se smeštaju u kružni bafer. Pamti se pokazivač na prvog kandidata za zamenu.
 - Kada u bafer putu zahtevana stranica ne postoji, a bafer je pun, onda se razmatra stranica koja je kandidat za zamenu.
 - Ako je njen refernce bit 1, njegova vrednost se postavlja na 0, a razmatranje za zamenu se vrši nad narednom stranicom (pokazivač se pomera na narednu stranicu).
 - Ako je refernce bit 0, onda se ta stranica menja novom, tj. u njen frejm se smešta nova.
-

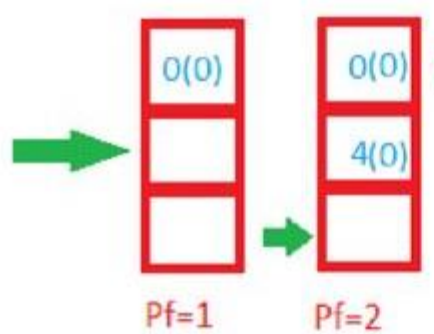
Clock (second chance)

Page sequence: **0** 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4



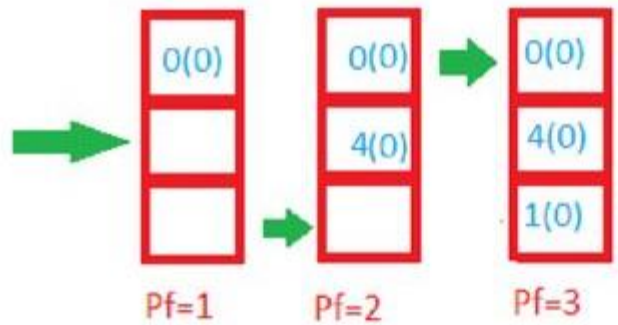
Clock (second chance)

Page sequence: **0 4** 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4



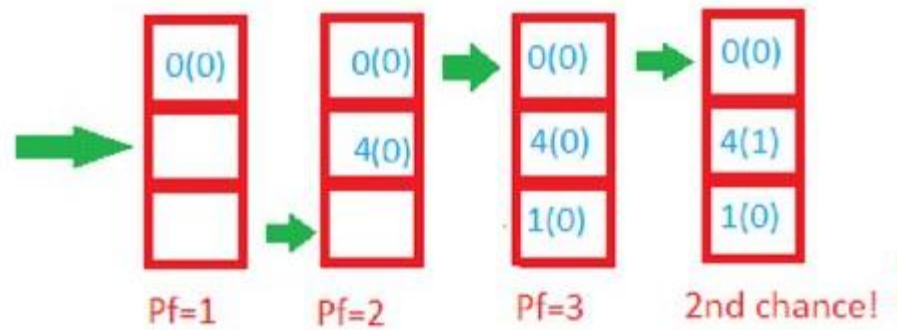
Clock (second chance)

Page sequence: **0 4 1** 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4



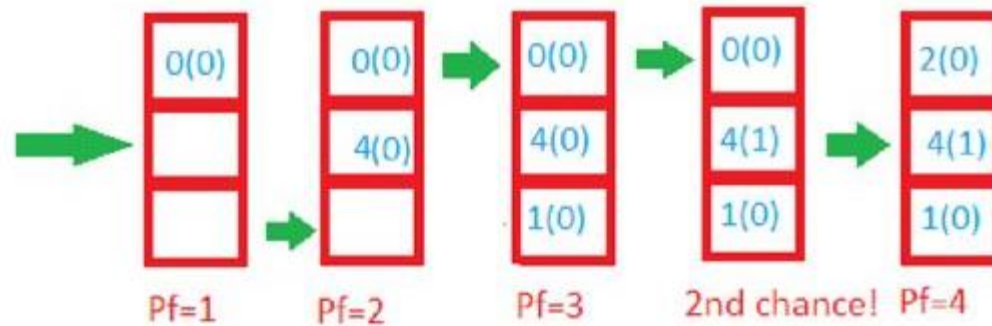
Clock (second chance)

Page sequence: **0 4 1 4** 2 4 3 4 2 4 0 4 1 4 2 4 3 4



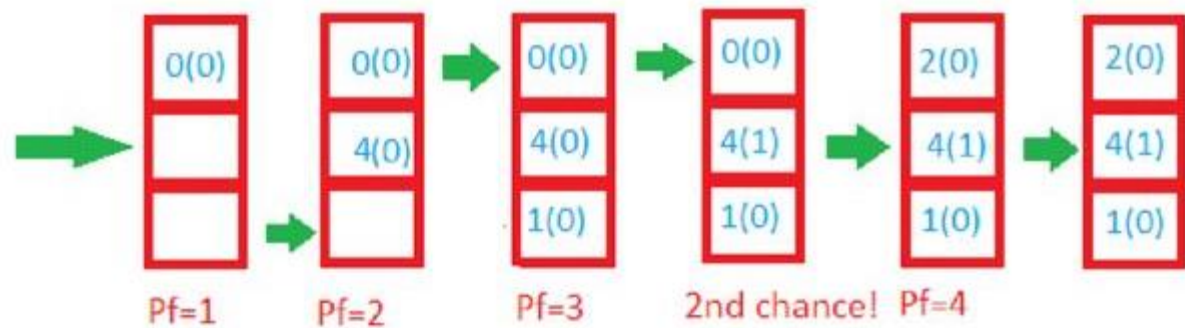
Clock (second chance)

Page sequence: **0 4 1 4 2** 4 3 4 2 4 0 4 1 4 2 4 3 4



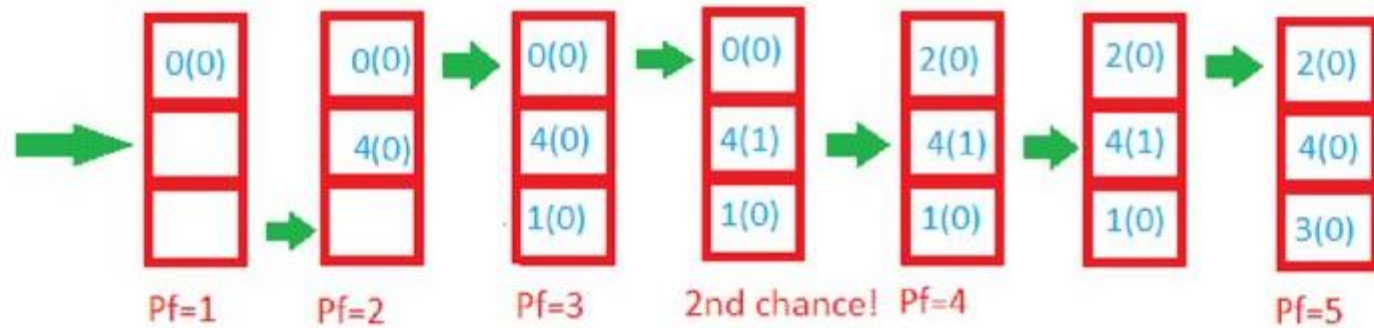
Clock (second chance)

Page sequence: **0 4 1 4 2 4** 3 4 2 4 0 4 1 4 2 4 3 4



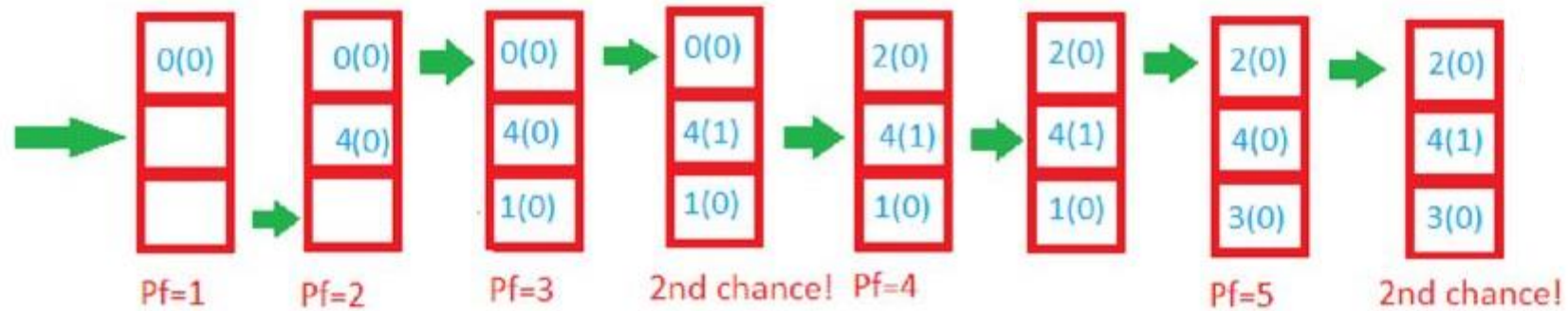
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3** 4 2 4 0 4 1 4 2 4 3 4



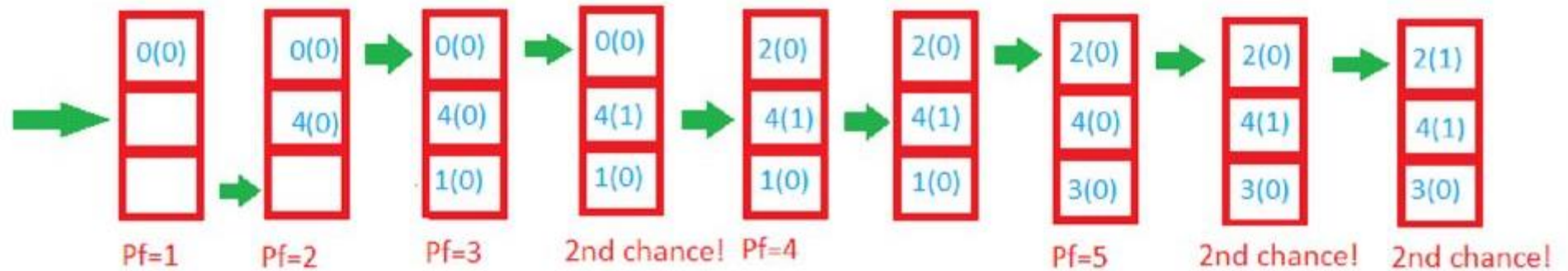
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4** 2 4 0 4 1 4 2 4 3 4



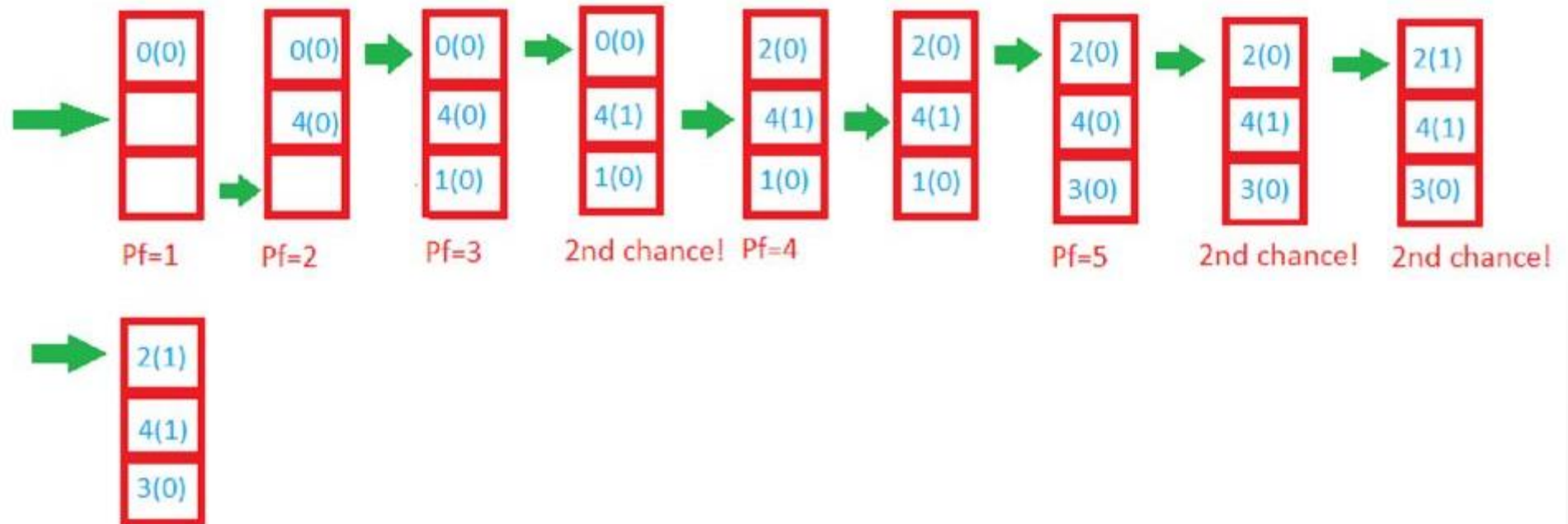
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2** 4 0 4 1 4 2 4 3 4



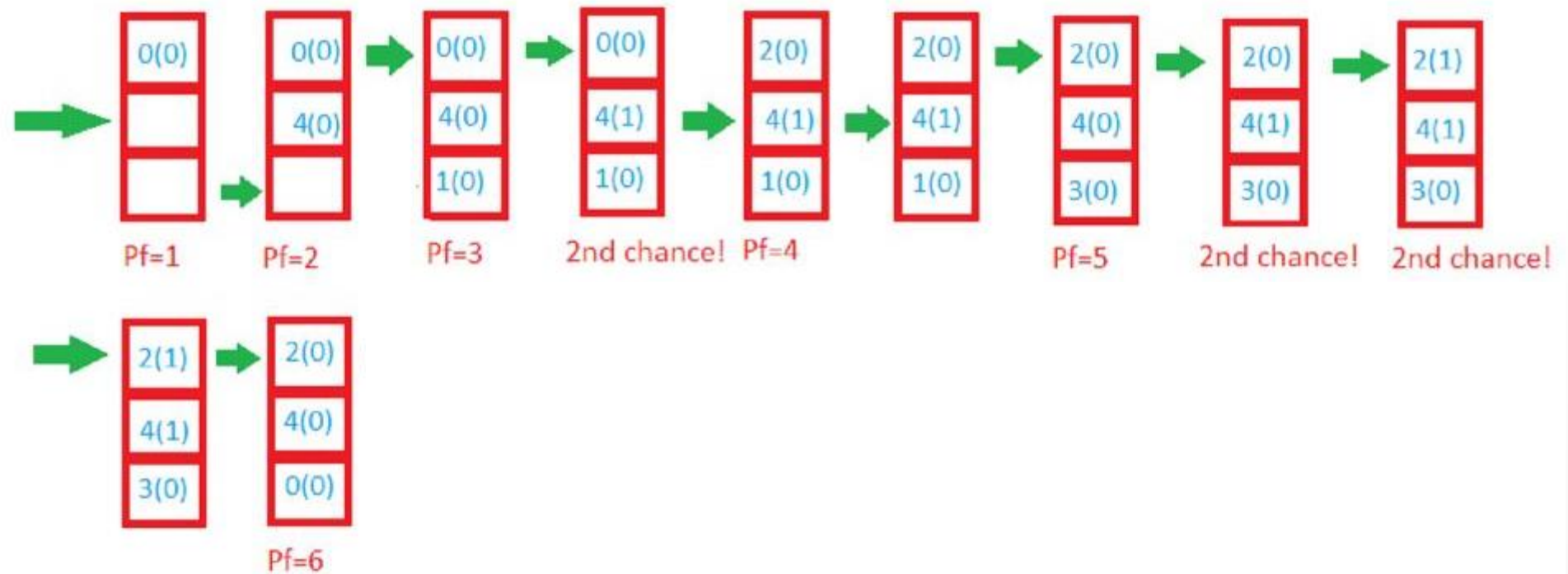
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4**



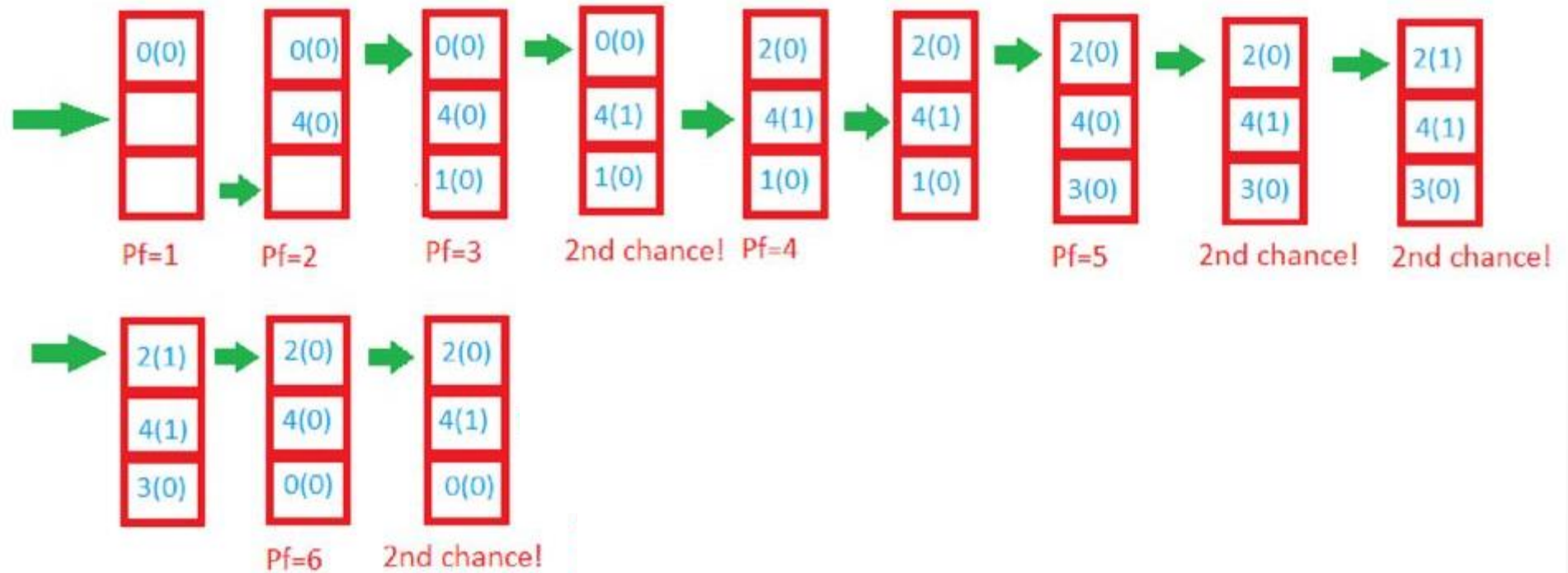
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0** 4 1 4 2 4 3 4



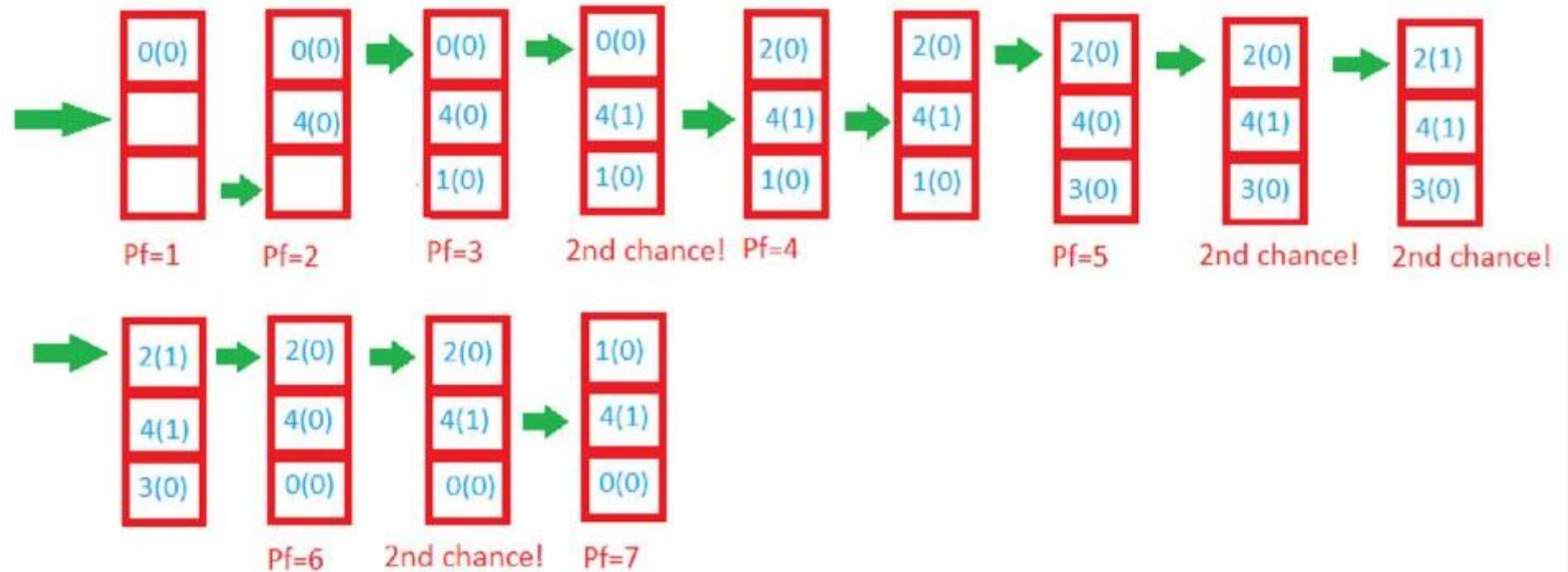
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4**



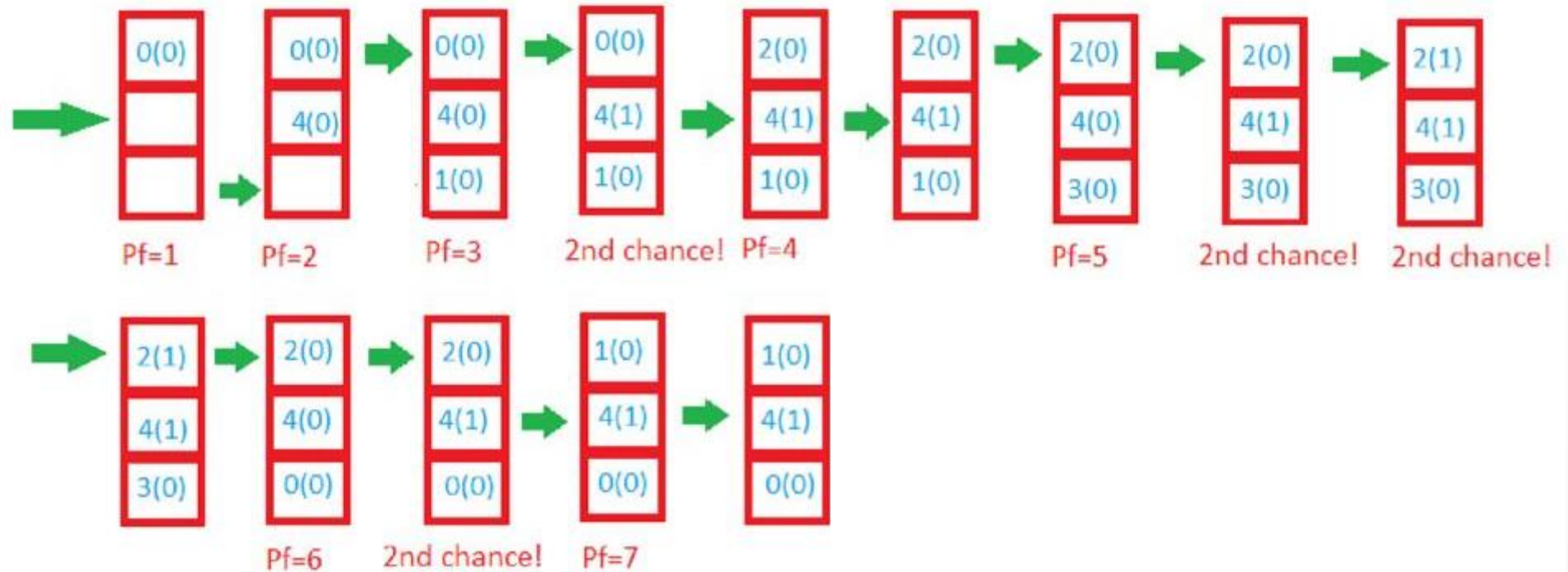
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4**



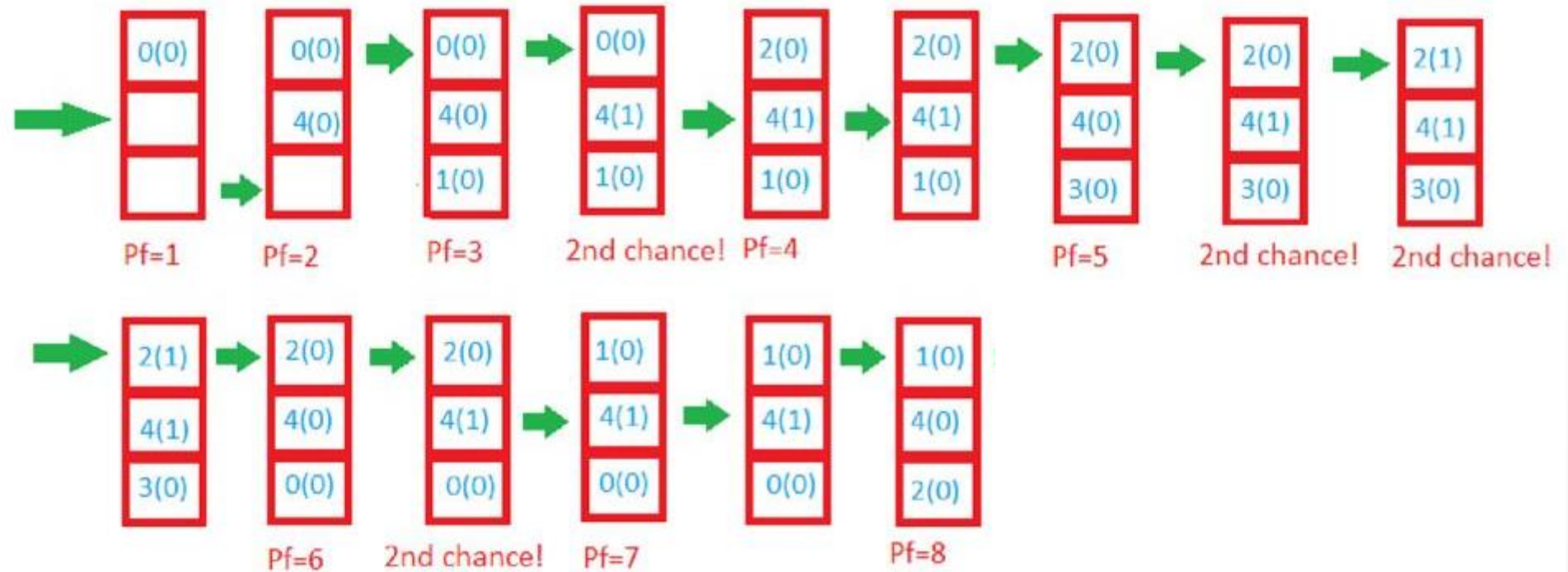
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4**



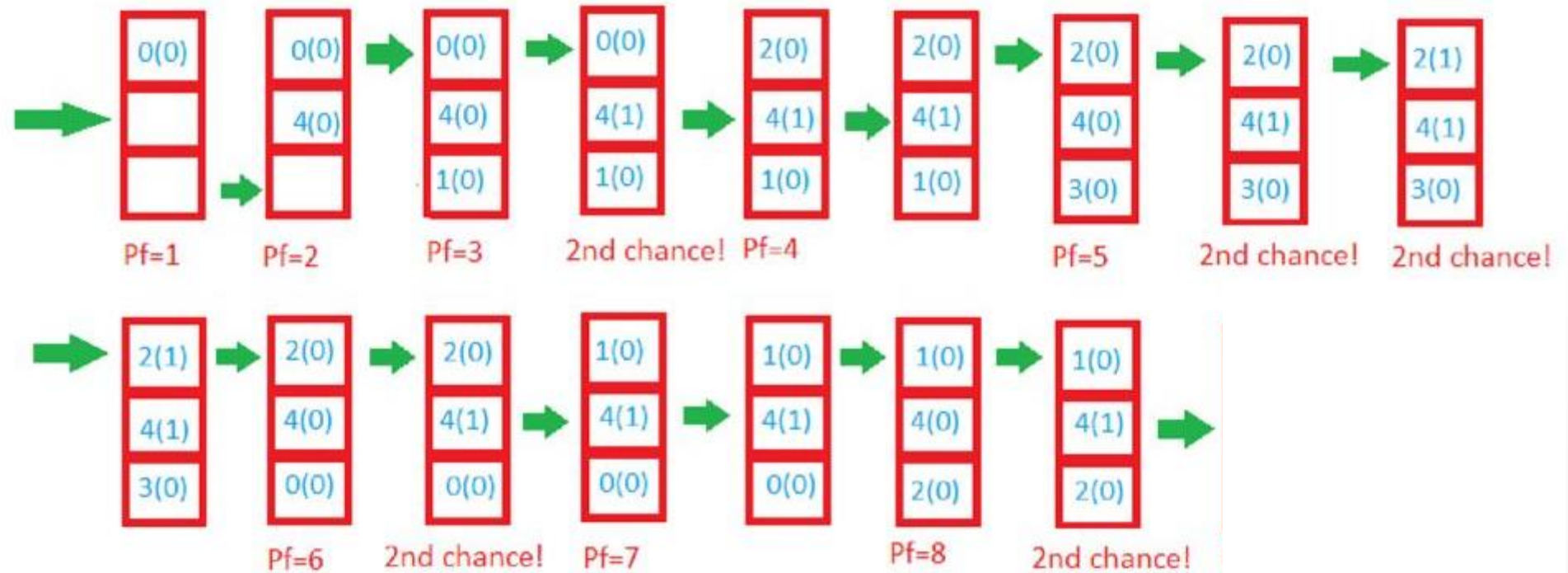
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4**



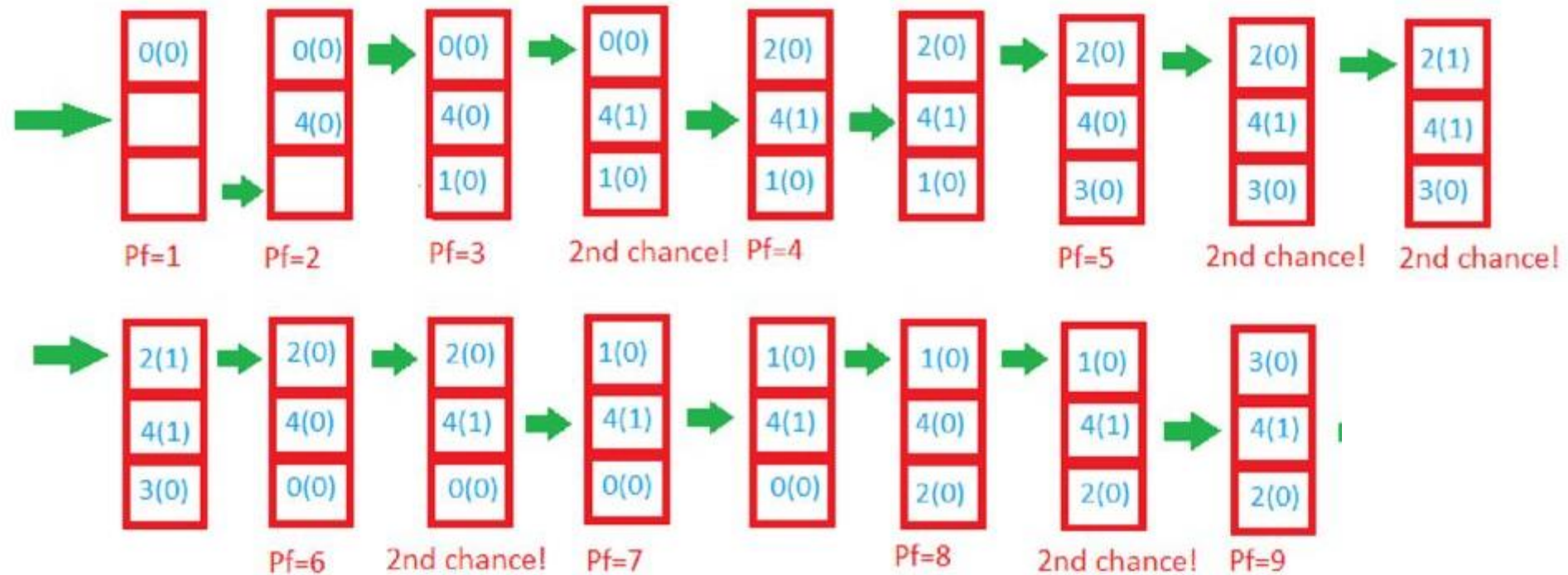
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4** 3 4



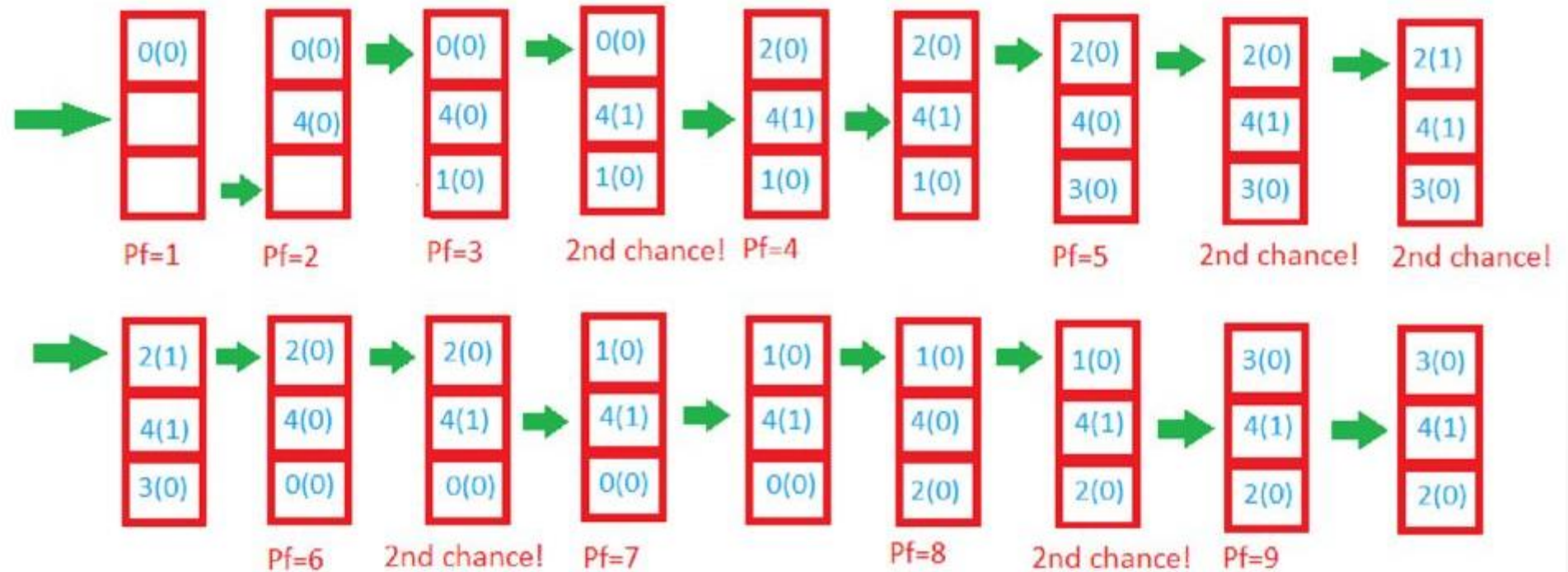
Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4**



Clock (second chance)

Page sequence: **0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4**



Sequential Flooding

A: a, b

B: c, d, e, f

for each record i in A

for each record j in B

do something with i and j

Sequential flooding

- each request –
page fault

LRU happens to
evict exactly the
page which we will
need next!

3 frames for A - enough	req.	a	a	a	a	b	b	b	b
		a	a	a	a	a	a	a	a
						b	b	b	b
3 frames for B < B		F				F			
	req.	c	d	e	f	c	d	e	f
		c	c	c	f	f	f	e	e
			d	d	d	c	c	c	f
				e	e	e	d	d	d
		F	F	F	F	F	F	F	F

LRU - K

- Na osnovu istorijskih podataka DMBS procenjuje kada će sledeći put određena stranica biti tražena.
 - Pamti se K vremenskih trenutaka pristupa i izračunava interval između dva uzastopna pristupa stranici.
-

Još neke tehnike

- Priority hints – DBMS na osnovu konteksta korišćenja stranica može da definiše njihov prioritet.
 - Localization – pamti se pristup stranicama koje se koriste za izvršavanje jednog upita (kružni bafer definisan za jedan upit).
-