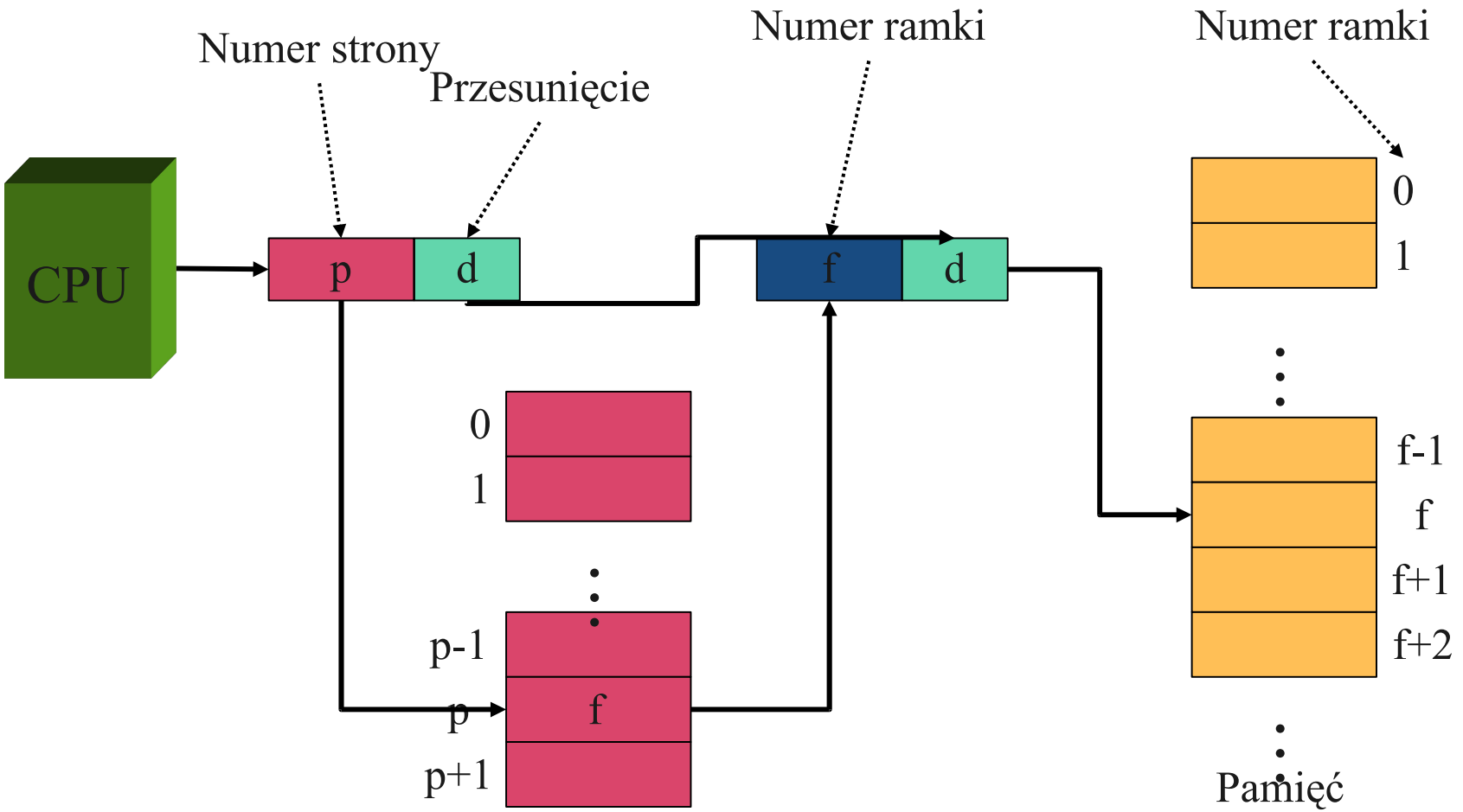
Wykład 8

Pamięć wirtualna

Wprowadzenie

- Podstawowa idea: System operacyjny pozwala na wykorzystanie pamięci o pojemności większej, od zainstalowanej pamięci RAM.
- Obszary logicznej przestrzeni adresowej, do których proces często się odwołuje przechowywane są w pamięci RAM.
- Obszary, do których proces odwołuje się rzadko, na dysku.
- Na dzisiejszym wykładzie skoncentrujemy się na ***stronicowaniu na żądanie*** (ang. on-demand paging).
- Mechanizm pamięci wirtualnej powinien być ukryty przed procesami użytkownika.
  - Proces "widzi" logiczną przestrzeń adresową od 0 do max_address
  - System operacyjny bez współpracy procesu przesyła dane do oraz z dysku.
- Pamięć wirtualna jest szczególnie przydatna w systemach wieloprogramowych.
  - Podczas gdy proces A jest zablokowany w oczekiwaniu na sprowadzenie danych z dysku, procesor może zostać przydzielony procesowi B.

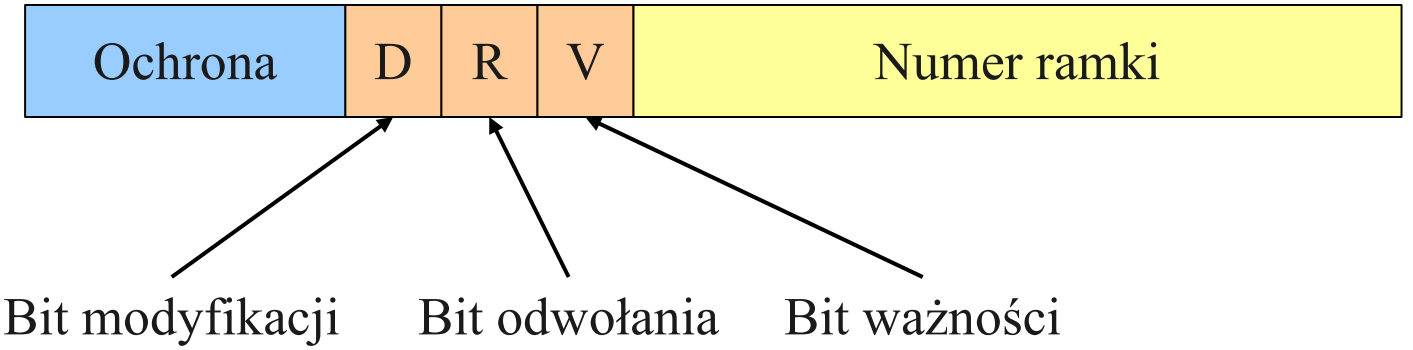
Przypomnienie: Translacja adresu w stronicowaniu



- Idea pamięci wirtualnej:

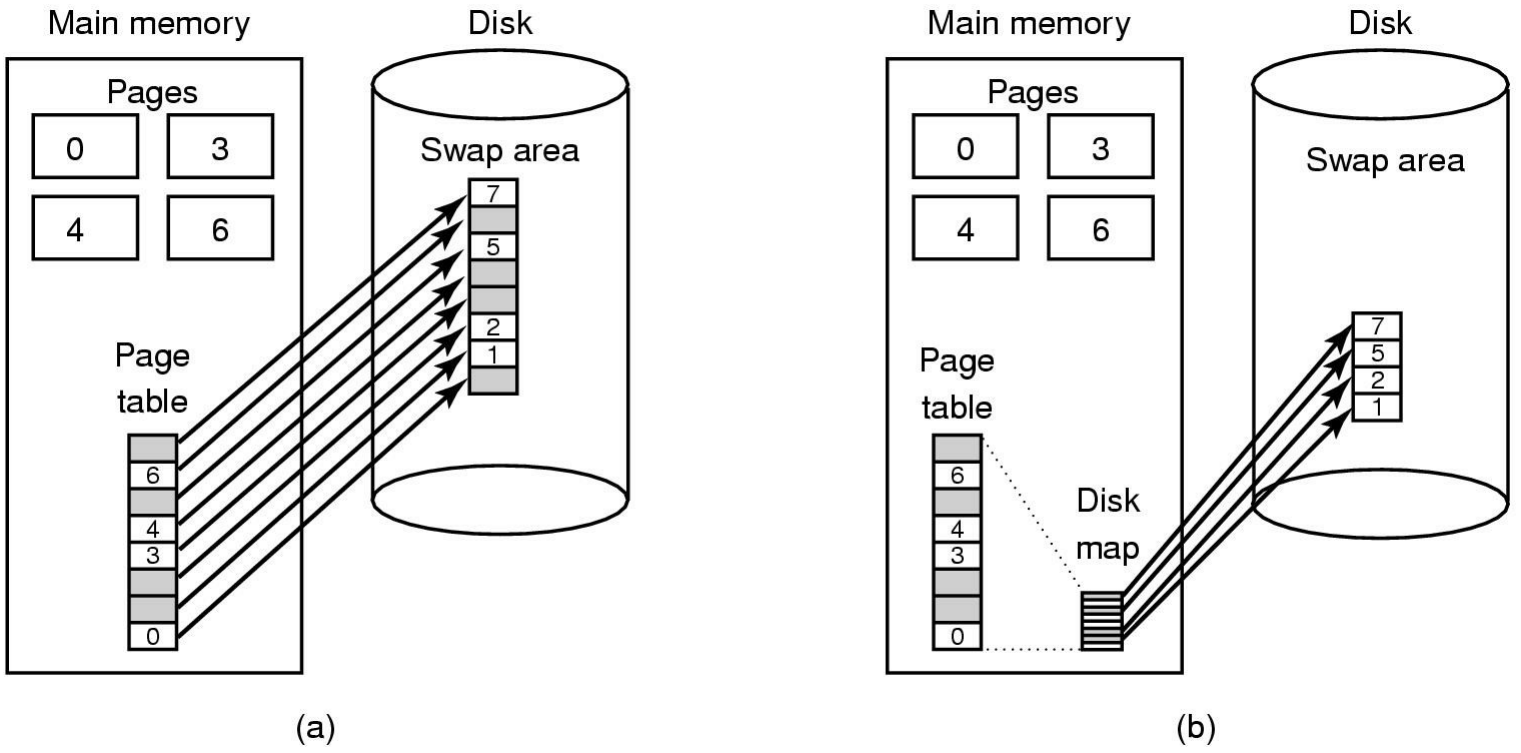
Przechowuj część stron w pamięci RAM, a część na dysku

Format pozycji tablicy stron



- V (ang. Valid) 1-strona jest w pamięci 0 – strony nie ma w pamięci.
  - Jeżeli V==0, to próba odwołania do strony generuje wyjątek ***błędu strony*** (ang. Page fault)
  - Jeżeli V==0, to numer ramki jest nieistotny. System operacyjny może w tym miejscu przechować numer bloku dyskowego, w którym zapisano stronę.
- Ochrona: Pozwolenie na zapis i odczyt (ewentualnie wykonanie).
- R (ang. Referenced) automatycznie ustawiany na jeden, jeżeli nastąpi odwołanie do strony.
- D (ang. Dirty) automatycznie ustawiany na jeden, jeżeli strona zostanie zmodyfikowana.
- R i D muszą być zerowane przez system !!!

Przechowywanie stron na dysku



- System operacyjny musi wiedzieć, gdzie znaleźć stronę.
  - (a) Ciągły obszar w przestrzeni wymiany odpowiada logicznej przestrzeni adresowej.
  - (b) Numer ramki w tablicy stron identyfikuje (być może przy pomocy tablicy pomocniczej) numer bloku dyskowego

Obsługa błędu strony (V==0)

- Błąd strony jest przerwaniem. Sprzęt musi umożliwiać wznowienie przerwanej instrukcji.
- System operacyjny sprawdza, czy chodzi o błędny adres, błąd ochrony, czy też o stronę której nie ma w pamięci.
  - Błędny adres lub błąd ochrony powoduje przerwanie procesu.
- Jeżeli strony nie ma w pamięci to należy:
  - Uśpić proces.
  - ***Znaleźć wolną ramkę*** **=> O tym więcej za chwilę.**
  - Wczytać stronę z dysku do ramki.
  - Wywołać planistę procesora.
  - Po pomyślnym odczycie strony Zmodyfikować pozycję w tablicy stron (numer ramki, V=1,D=0,R=0)
  - Wznowić proces.

Wpływ na wydajność

- Czas dostępu do pamięci RAM: 100 ns.
- Czas sprowadzenia strony z dysku: 10 ms.
- Błąd braku strony zwiększa czas dostępu 100 000 razy

- Wniosek: Powinniśmy zadbać o to, aby błędy braku stron występowały ***bardzo rzadko.***

Zastępowanie stron
(Co robimy, gdy nie ma wolnej ramki)

- Przyjmijmy że procesowi przydzielono k ramek pamięci oraz że wszystkie ramki są już zajęte.
- W takiej sytuacji przy błędzie braku strony musimy wybrać jedną ze stron obecnych w pamięci (***stronę-ofiarę***)
  - Jeżeli ta strona została zmodyfikowana (D==1), to należy ją zapisać na dysk.
  - Ramka odpowiadająca wybranej stronie jest zwalniana.
- Problem: jak wybrać stronę-ofiarę ?
  - Zazwyczaj chcemy, aby strony, do których proces się często odwołuje, przebywały na stale w pamięci.
  - Jak określić “częstość” przy ograniczonych możliwościach sprzętowych ?
- Wybór strony-ofiary jest wykonywany przez algorytm zastępowania stron (ang. page replacement).

Porównanie algorytmów zastępowania stron.

- Kryterium: Minimalizacja liczby błędów stron.
- Dla porównania przyjmijmy, że proces się odwołuje kolejno do stron:

1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5.

Algorytm FIFO (ang. First In First Out).

- Zastąp stronę, która została sprowadzona jako pierwsza do pamięci. Sprowadzone strony tworzą kolejkę kolejkę.
- Kolejność odwołań: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5.
- Jeżeli procesowi przydzielono trzy ramki:

1	1	4	5	
2	2	1	3	9 błędów stron
3	3	2	4	

- Jeżeli procesowi przydzielono cztery ramki:

1	1	5	4	
2	2	1	5	10 błędów stron
3	3	2		
4	4	3		

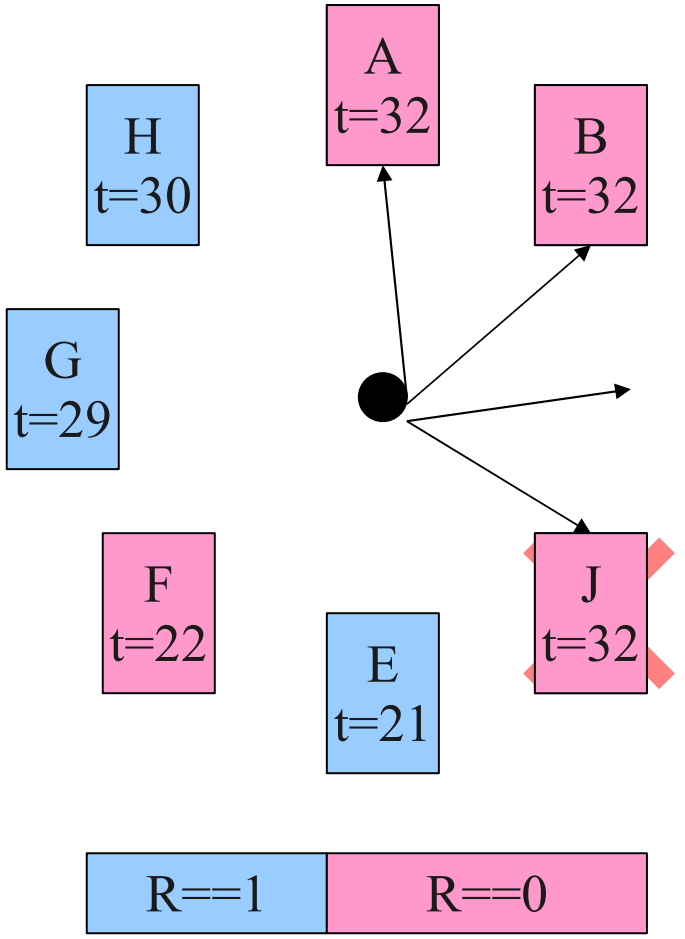
- Anomalia Belady'ego: większa liczba ramek niekoniecznie zmniejsza liczbę błędów stron.

Algorytm drugiej szansy

- Jest to modyfikacja algorytmu FIFO.
- W standardowym algorytmie FIFO wybierana jest pierwsza strona z kolejki.
- W algorytmie drugiej szansy sprawdzany jest bit odniesienia R.
  - Jeżeli R==0 (brak odniesienia) to strona jest wybierana na ofiarę.
  - Jeżeli (R==1) (odniesienie) to:
    - R ustawiany jest na zero
    - Strona przesunięta jest na koniec kolejki (“otrzymała drugą szansę”).
    - Przechodzimy do kolejnej strony w kolejce.

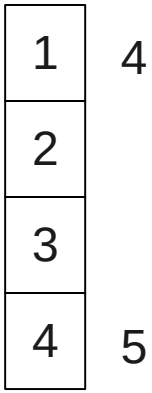
Algorytm zegarowy

- Inne sformułowanie algorytmu drugiej szansy.
- Wskazówka zegara wskazuje na stronę wybraną do zastąpienia.
- W przypadku błędu braku strony:
  - Jeżeli R==0 to zastąp tę stronę
  - Jeżeli R==1 to ustaw R==0 i przesuń dalej wskazówkę.



Algorytm optymalny

- Zastąp stronę, do której nie będziemy się odwoływać przez najdłuższy czas
- Kolejność odwołań: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5.

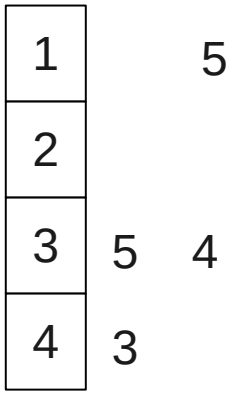


6 błędów strony

- Algorytm niemożliwy do zaimplementowania w praktyce.
- Stosowany do porównywania innych algorytmów.

Algorytm LRU (Least recently used)

- Zastąp tę stronę, która nie była najdłużej używana. Zakładamy, że ta strona nie będzie dalej potrzebna
- Brak anomalii Belady'ego.
- Kolejność odwołań: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5.

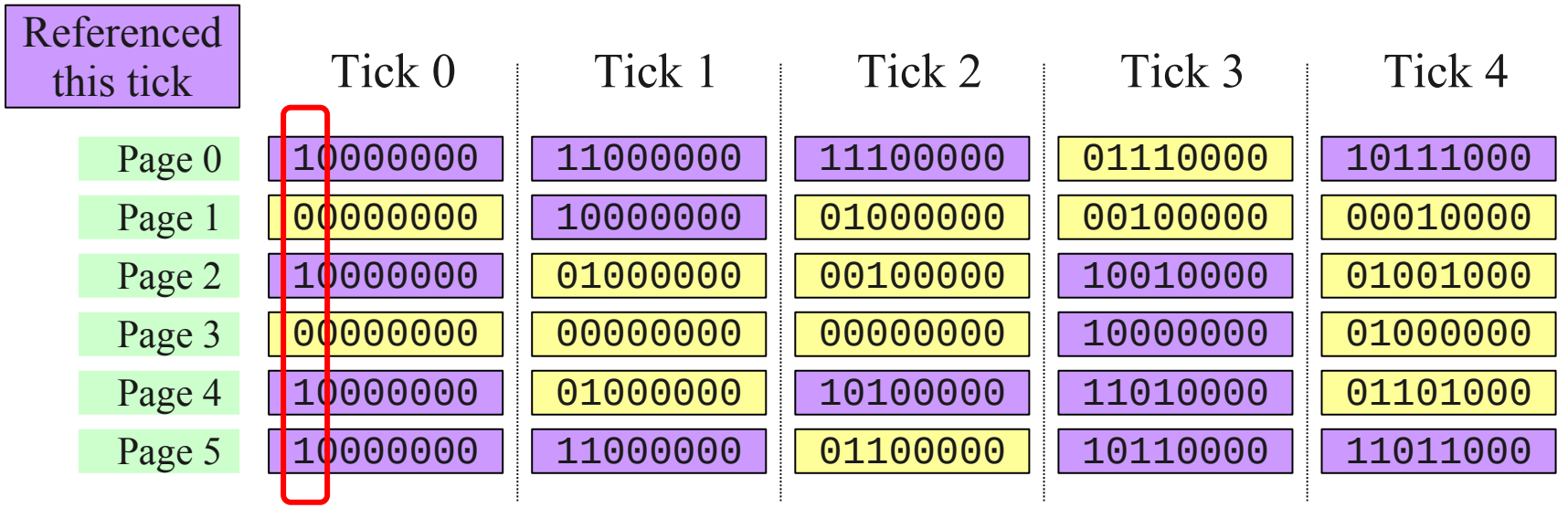


- Niezły algorytm, ale nadal trudny do zaimplementowania.
  - Większość systemów ma tylko bity R oraz V.
- W praktyce stosuje się algorytmy przybliżające zasadę LRU.

Symulowanie algorytmu LRU

- Aproksymacja poprzez algorytm NFU (Not Frequently Used).
- Co pewien czas, w procedurze obsługi, przerwania zegarowego system przegląda tablicę stron.
- Dla strony w której R==1 zwiększana jest zawartość licznika i R jest ustawiany na zero.
- W ten sposób strony, do których występuje wiele odwołań, charakteryzują się dużą wartością licznika.
- Jako ofiara wybierana jest strona o największej wartości licznika.
- Problem: Strona była często używana 10 sekund temu, obecnie nie jest już używana (Wysoka wartość licznika pozostaje).

Postarzanie (ang. aging)



- Idea: Wraz z upływem czasu zmniejszaj wartości liczników.
- Licznik zajmuje 8 bitów.
- W każdym cyklu przesuń zawartość wszystkich liczników w prawo (podziel przez dwa).
- Jeżeli było odwołanie to ustaw najstarszy bit na jeden. Zatem algorytm wygląda tak:

```
counter/=2
if (referenced) counter+=128
```

Pobieranie stron do pamięci

- Pobieranie w momencie wystąpienia błędu braku strony.
- Jeżeli strona jest tylko do odczytu, np. kod programu to możemy wczytać ją z pliku wykonywalnego.
  - Przy starcie procesu żadna strona nie jest w pamięci.
  - Duża liczba błędów strony przy starcie programu.
  - Fragmenty kodu, które nie są wykorzystane, nie zostaną wczytanie do pamięci.
- Wczytuj strony grupami.
  - Jeżeli mamy odwołanie do strony a, to prawdopodobne są odwołanie do stron a+1,a+2,...

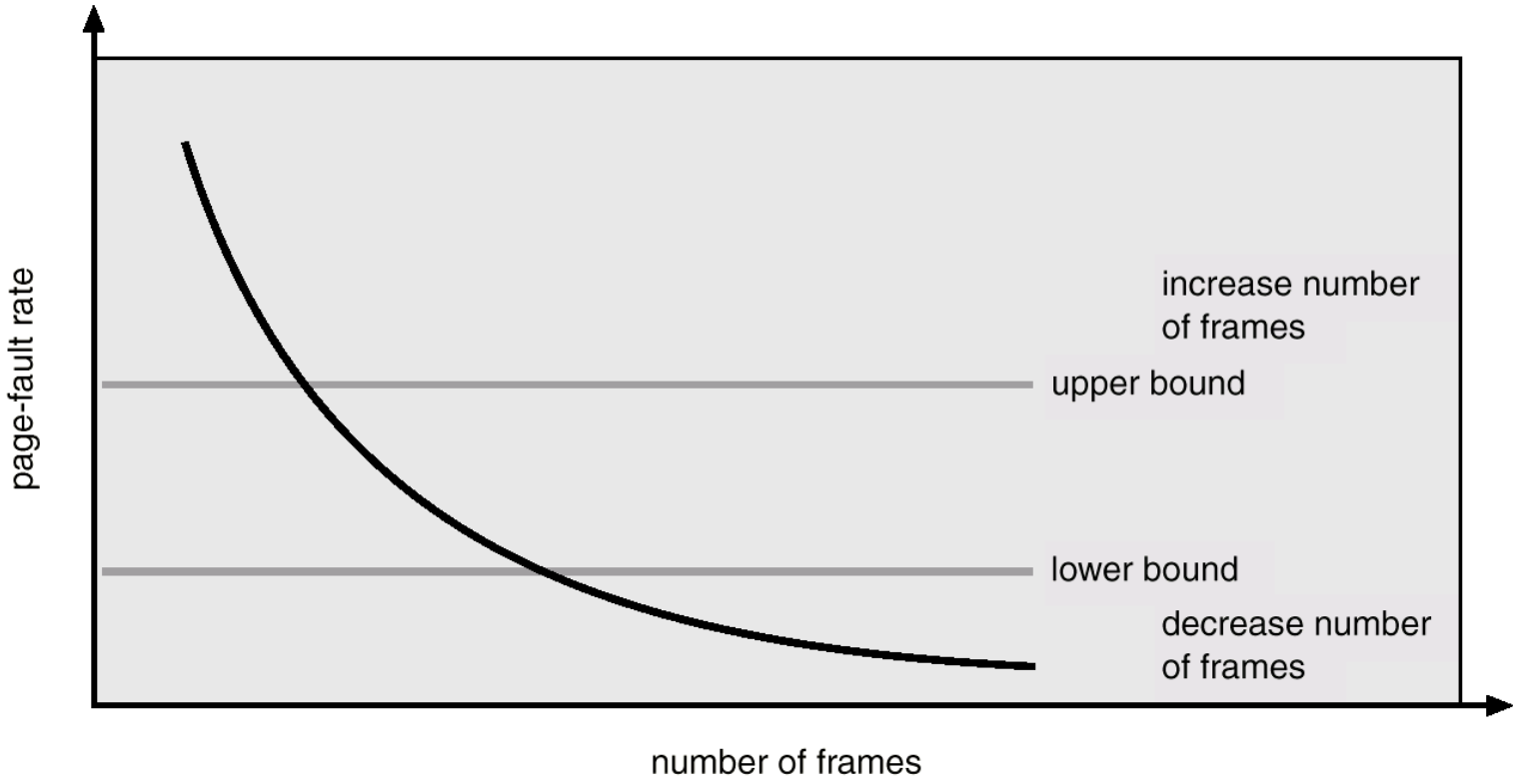
Zapisywanie zmodyfikowanych stron

- Strona zapisywana w momencie jej zastąpienia.
  - Mała liczba zapisów na dysk
  - Algorytm powolny: brak strony powoduje konieczność zapisania strony na dysk i wczytania strony z dysku.
- Strony zapisywane periodycznie w tle.
  - Proces drugoplanowy przegląda strony i zapisuje strony zmodyfikowane (D==1), do których ostatnio nie było odwołań.
  - Zapisanie strony powoduje skasowanie bitu D.
  - Proces drugoplanowy może zapisywać strony grupami => większa wydajność operacji dyskowych.

Stronicowanie na żądanie w systemie wieloprogramowym

- Jak dotąd nie uwzględnialiśmy faktu współistnienia wielu procesów
- Zastępowanie stron w systemie wieloprogramowym.
  - Lokalne: strona-ofiara jest wybrana wyłącznie spośród stron procesu.
  - Globalne: strona-ofiara jest wybrana spośród stron wszystkich procesów
- Problem: Jak rozdzielić dostępne ramki pomiędzy procesy
  - Przydziel stałą liczbę ramek: 100 ramek, 5 procesów, przydziel po 20 ramek.
  - Przydziel liczbę ramek proporcjonalną do rozmiaru procesu.

Dynamiczna alokacja ramek



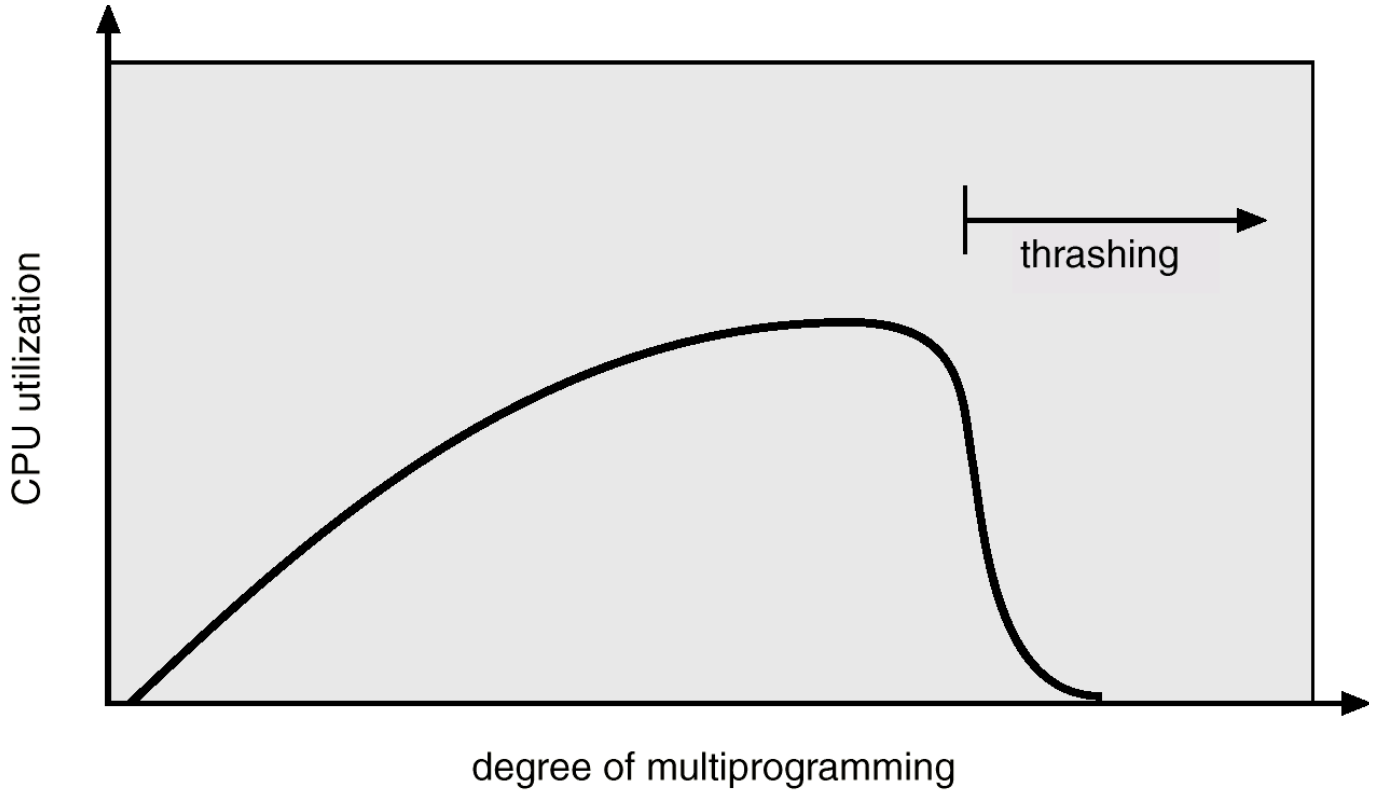
- Obserwuj częstość błędów stron.
  - Jeżeli zbyt dużą, to przydziel dodatkowe ramki
  - Jeżeli zbyt mała, to zabierz ramki

Buforowanie stron

- Zastępowanie stron niekoniecznie musi być przeprowadzane w momencie wystąpienia błędu braku strony.
- Odzyskane strony są umieszczane na jednej z dwóch list.
  - Lista stron zmodyfikowanych
  - Lista stron niezmodyfikowanych – bufor stron (ang. page buffer)
- Strony z listy stron zmodyfikowanych są grupami zapisywane na dysk i przenoszone do listy stron niezmodyfikowanych stron
- W przypadku zapotrzebowania na nową ramkę przydzielana jest jedna ramka
- Z bufora stron.
- W przypadku wystąpienia błędu braku stron sprawdzamy, najpierw sprawdzamy czy ta strona jest na jednej z dwóch list.

Szamotanie (ang. trashing)

- Jeżeli liczba ramek jest zbyt mała liczba błędów braku strony jest bardzo duża. Procesy ciągle oczekują na sprowadzenie stron z dysku i na dysk.
  - Wykorzystanie procesora jest bardzo małe. System (wsadowy) może zdecydować o uruchomieniu dodatkowych procesów, co dodatkowo zmniejszy liczbę wolnych ramek i zwiększy szamotanie.



Pamięć wirtualna, a we-wy

- Proces P1 zgłasza żądanie odczytu z we-wy do bufora w swojej pamięci.
  - Czy strony, w których znajduje się bufor, są w pamięci ? Jeżeli nie, to strony trzeba sprowadzić z dysku.
- Gdy P1 czeka na wykonanie odczytu, wykonuje się P2
- P2 zgłasza błąd strony.
- Problem: do zastąpienia może być wybrana strona P1, w której znajduje się bufor.
- Blokowanie (ang. locking) stron.
  - Strona zablokowana nie może być wybrana do zastąpienia.
  - Strony powinny pozostawać zablokowane przez stosunkowo krótki czas.

Optymalizacja kodu

- Array A[1024, 1024] of integer, A[*wiersz*,*kolumna*]
- Każdy wiersz (4KB) jest przechowany na jednej stronie
- Procesowi przydzielono jedną ramkę (4KB)
  - Program 1

```
for j := 1 to 1024 do
    for i := 1 to 1024 do
        A[i,j] := 0;
```

    1024 x 1024 błędów strony
  - Program 2

```
for i := 1 to 1024 do
    for j := 1 to 1024 do
        A[i,j] := 0;
```

    1024 błędy strony
- Należy preferować kod odwołujący się do komórek pamięci o kolejnych adresach. (również ze względu na optymalne wykorzystanie pamięci podręcznej i szyny procesora).