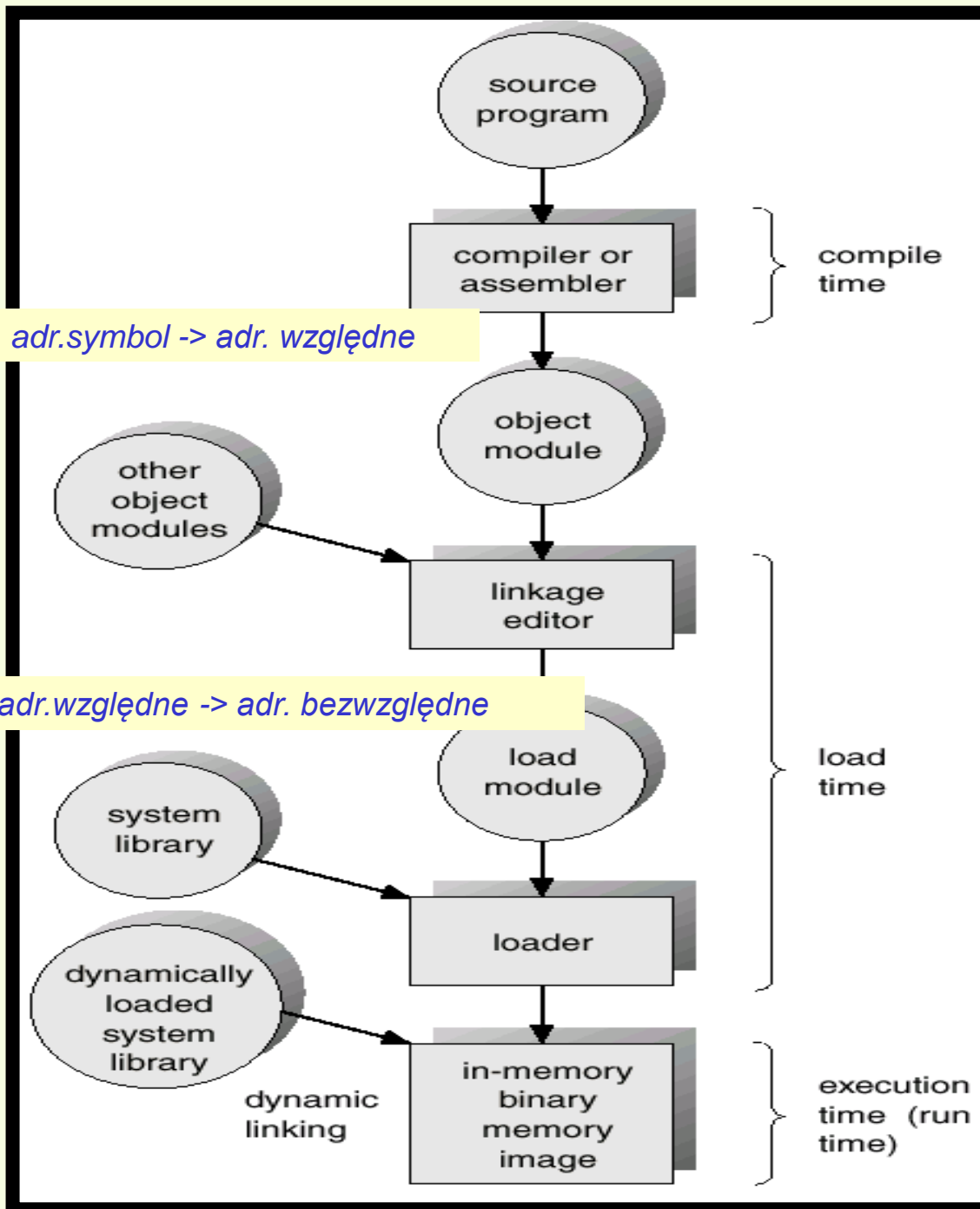


ZARZĄDZANIE PAMIĘCIĄ OPERACYJNĄ

Wiązanie adresów

->adresy pamięci

- kompilacja;
–kod bezwzględny (*.com)
- ładowanie;
–kod przemieszczalny
- wykonanie



Optymalizacja wykorzystania pamięci

- Ładowane dynamiczne

(podprogram ładowany w momencie wywołania)

- Konsolidacja dynamiczna

(w obrazie binarnym stub – namiastka procedury wskazująca jak odnaleźć podprogram biblioteczny)

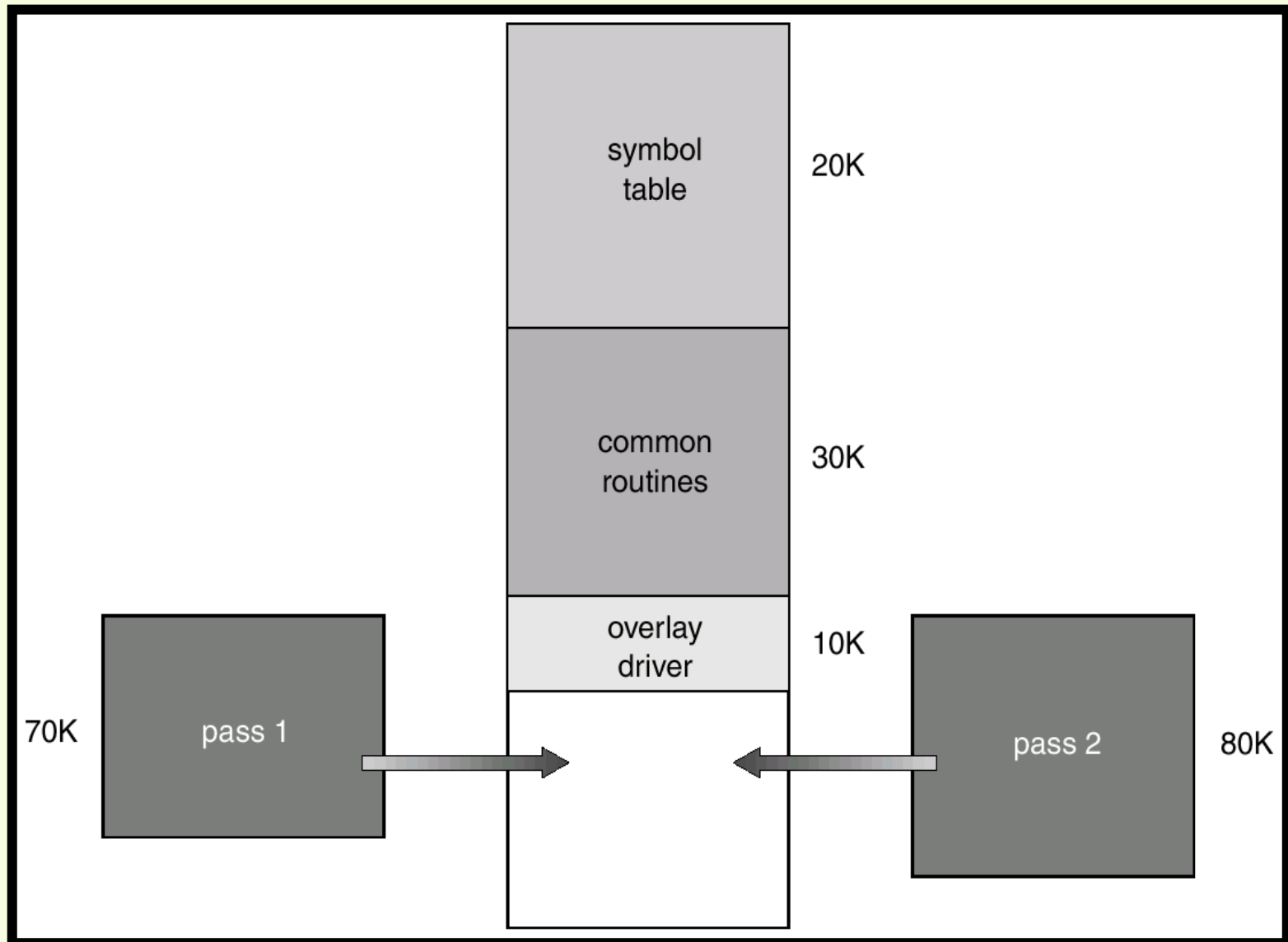
- Nakładki

(zawierają moduły konieczne w danym momencie;
projektowane przez programistę)

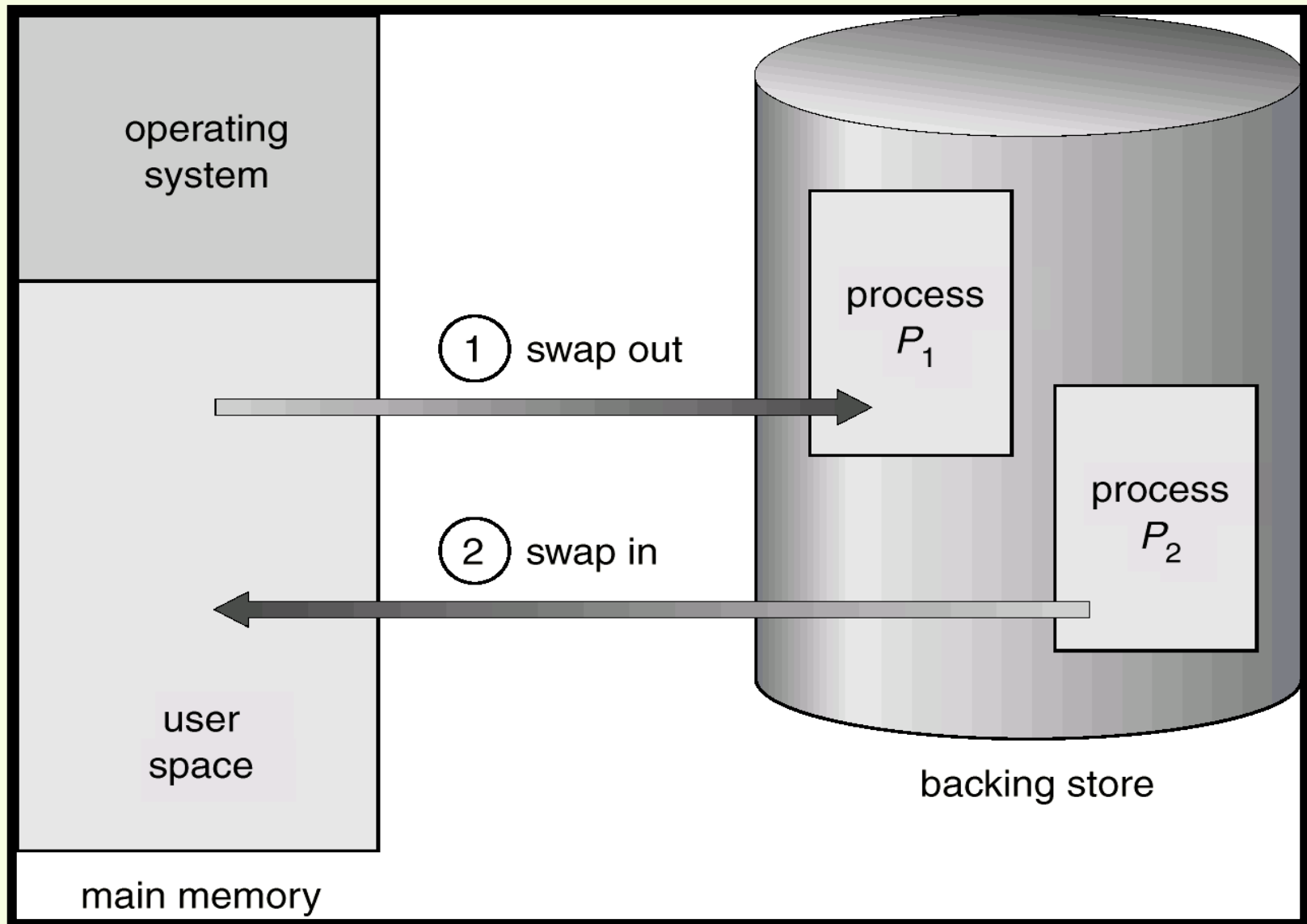
- Wymiana

(rotacyjny, priorytetowy alg. planowania;
roll in roll out ; ekspedytor – *dispatcher*;
długi czas przełączania kontekstu – 200ms;
kwant czasu procesora – 0.2 s)

Nakładki - dwuprzebiegowy assembler



swapping



- adres logiczny –

- wytworzony przez procesor (adres wirtualny)

- zbiór wszystkich adresów logicznych

- logiczna przestrzeń adresowa

- adres fizyczny –

- umieszczony w rejestrze adresowym pamięci

- zbiór wszystkich adresów fizycznych –

- fizyczna przestrzeń adresowa

- odwzorowanie adresów wirtualnych na fizyczne

- MMU - jednostka zarządzająca pamięcią

- *-memory-management-unit)*

- Ustalanie adresów podczas kompilacji i ładowania
=> adresy logiczne i fizyczne są takie same
- Ustalanie adresów podczas wykonania =>
adresy logiczne i fizyczne są różne

adres logiczny=adres wirtualny

Przydział ciągły - pojedynczy obszar

Rejestry: bazowy (RB) + rejestr graniczny (RG)

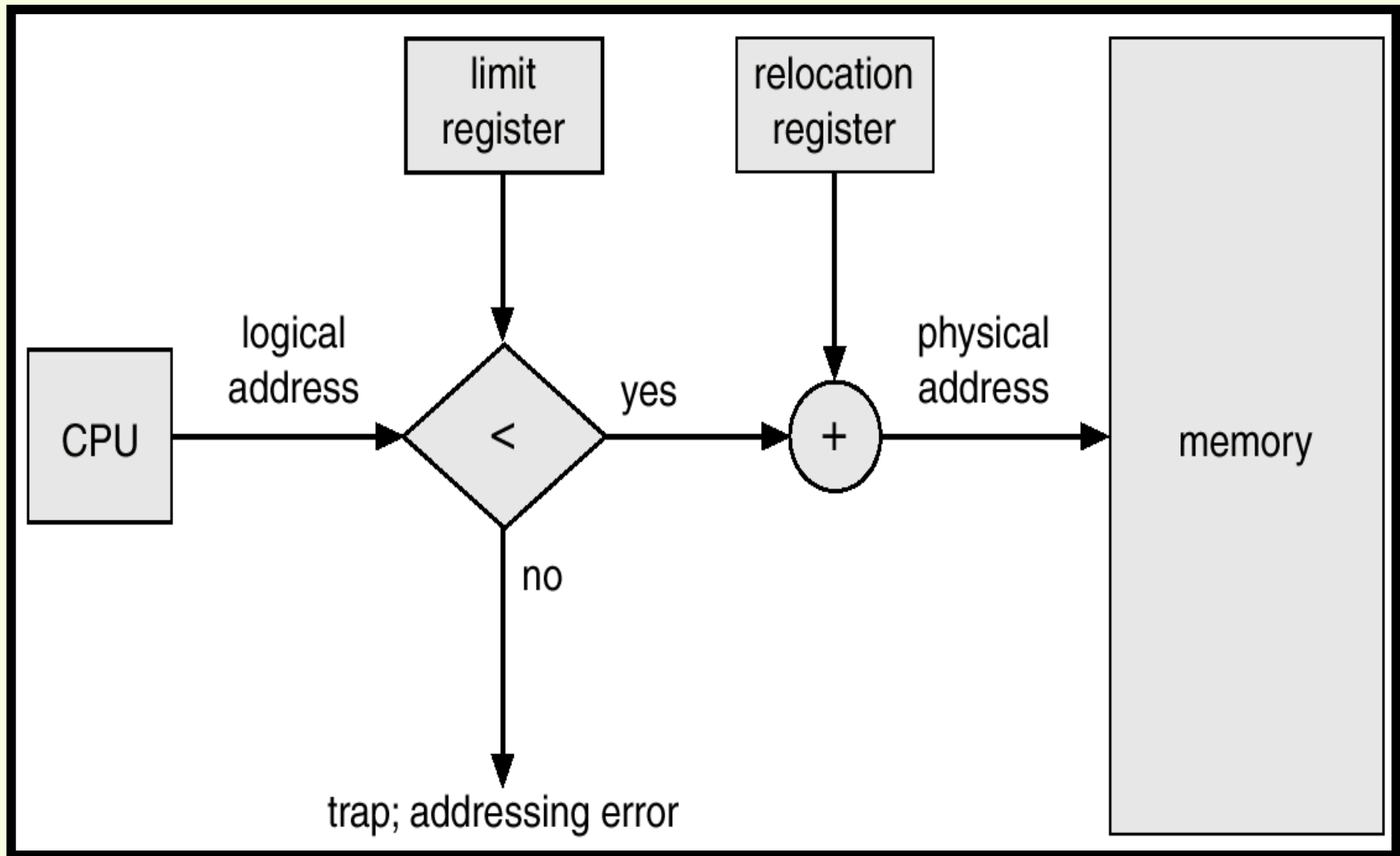
if $a < 0$ then przekroczony zakres pamięci

$a' := RB + a$

if $a' > RG$ then przekroczony zakres pamięci

a' jest żądanym adresem komórki pamięci

- przestrzeń adresów odwzorowywana zgodnie z tym schematem jest **liniowa**, a jej rozmiar - stanowiący różnicę między RB i RG
- nie może być większy niż rozmiar przestrzeni pamięci



Przydzielanie wielu obszarów

- **MFT** (multiprogramming with a fixed number of tasks)

podział pamięci na obszary o stałym rozmiarze, przydzielane procesom
ogranicza wieloprogramowość

- **MVT** (multiprogramming with a variable number of tasks)

środowisko wsadowe; lista wolnych dziur;
problem dynamicznego przydziału pamięci:

- pierwsze dopasowanie (*first fit*)

- najlepsze dopasowanie

- najgorsze dopasowanie

- fragmentacja zewnętrzna**

(reguła 50% - ff; na N przydzielonych bloków ginie N/2)

- fragmentacja wewnętrzna**

- nakład na trzymanie informacji o małych dziurach
 - przekracza ich wartość - dołączanie do większych przydziałów

Stronicowanie pamięci

- Rozwiązane problemu zewnętrznej fragmentacji
- Nieciągła logiczna przestrzeń adresowa procesów
- Pamięć fizyczna – podzielona na ramki stałej wielkości
- Pamięć logiczna – podzielona na strony o tym samym rozmiarze

Stronicowanie pamięci

- Około roku 1960 w Uniwersytecie Manchesterskim wprowadzono pojęcie pamięci jednopoziomowej

- Pole adresowe – 16 bitów

- Teoretyczna pamięć $2^{16} = > < 0; 65535 >$

- Pamięć fizyczna – 4096 słów

- Pamięć jednopoziomowa –

pamięć pomocnicza stanowi rozszerzenie pamięci głównej

Zadania mechanizmu stronicowania

- odwzorowywanie adresów
 - określanie, do której strony odnosi się adres w programie oraz znajdowanie (o ile taka istnieje) ramki strony, którą bieżąco zajmuje dana strona
- przesyłanie - w zależności od potrzeby
 - stron z pamięci pomocniczej do pamięci głównej
 - oraz odsyłanie nie używanych już stron z powrotem z pamięci głównej do pomocniczej.

Adres logiczny

- bardziej znaczące bity - numer strony
- mniej znaczące zaś bity - numer bajtu na stronie (*offset*)
- rozmiar strony = 2^n
 - n** mniej znaczących bitów adresu - numer bajtu,
pozostałe bity oznaczają numer strony
- liczba bitów w adresie wystarcza do zaadresowania całej pamięci wirtualnej.

Przykład

adres składa się z **32** bitów

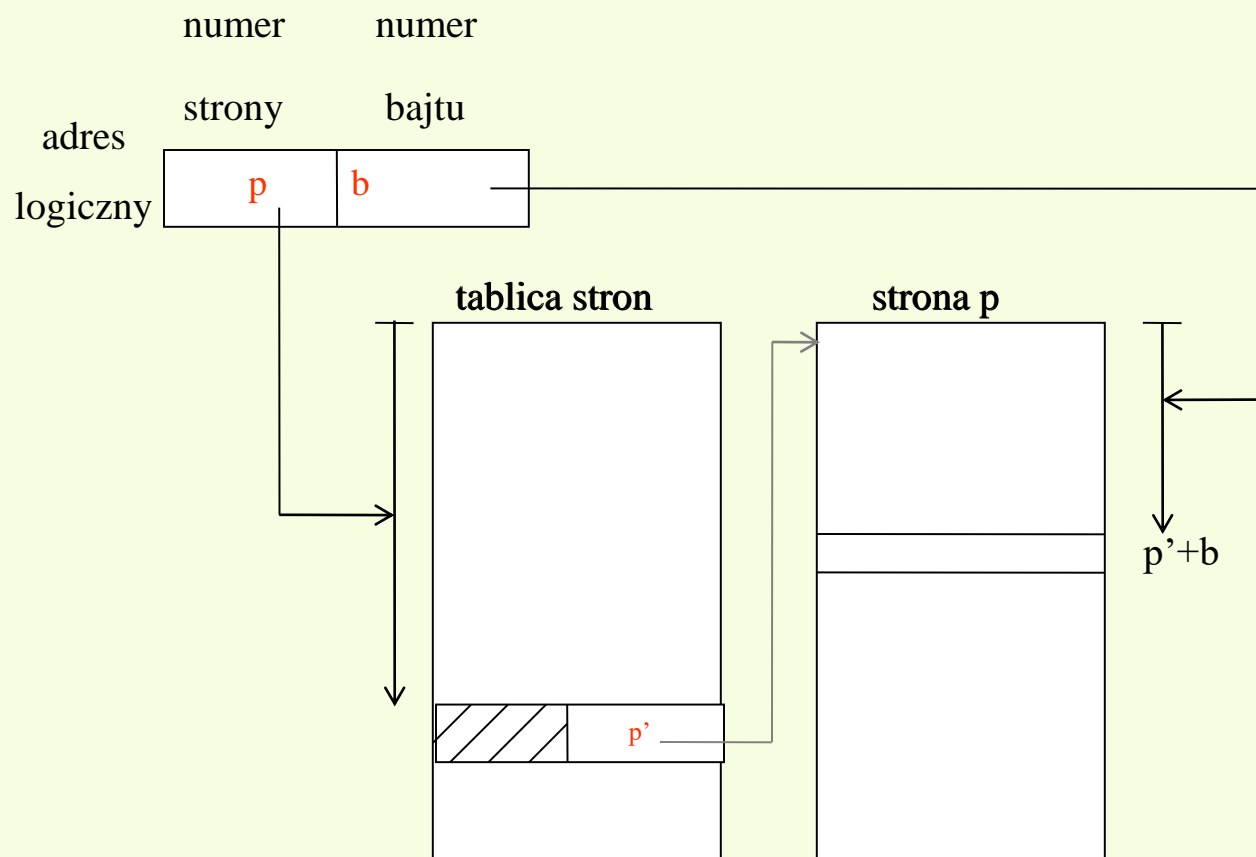
pamięć wirtualna może mieć 2^{32} bajtów
rozmiar strony wynosi **512** bajtów, czyli 2^9 .

9 mniej znaczących bitów - numer bajtu,
23 bardziej znaczące bity - numer strony

- podział adresu na numery stron i bajtów
 - wykonywany sprzętowo;
 - niewidoczny dla programisty
 - programista dysponuje dużą sekwencyjną przestrzenią adresów.

• odwzorowanie adresu logicznego na adres komórki w pamięci fizycznej dokonuje się za pomocą tablicy stron, w której element p zawiera adres p' ramki strony zawierającej stronę numer p .

odwzorowanie adresu przy użyciu tablicy stron



odwzorowanie adresu przy użyciu tablicy stron

Odwzorowanie adresu jest zatem określone wzorem:

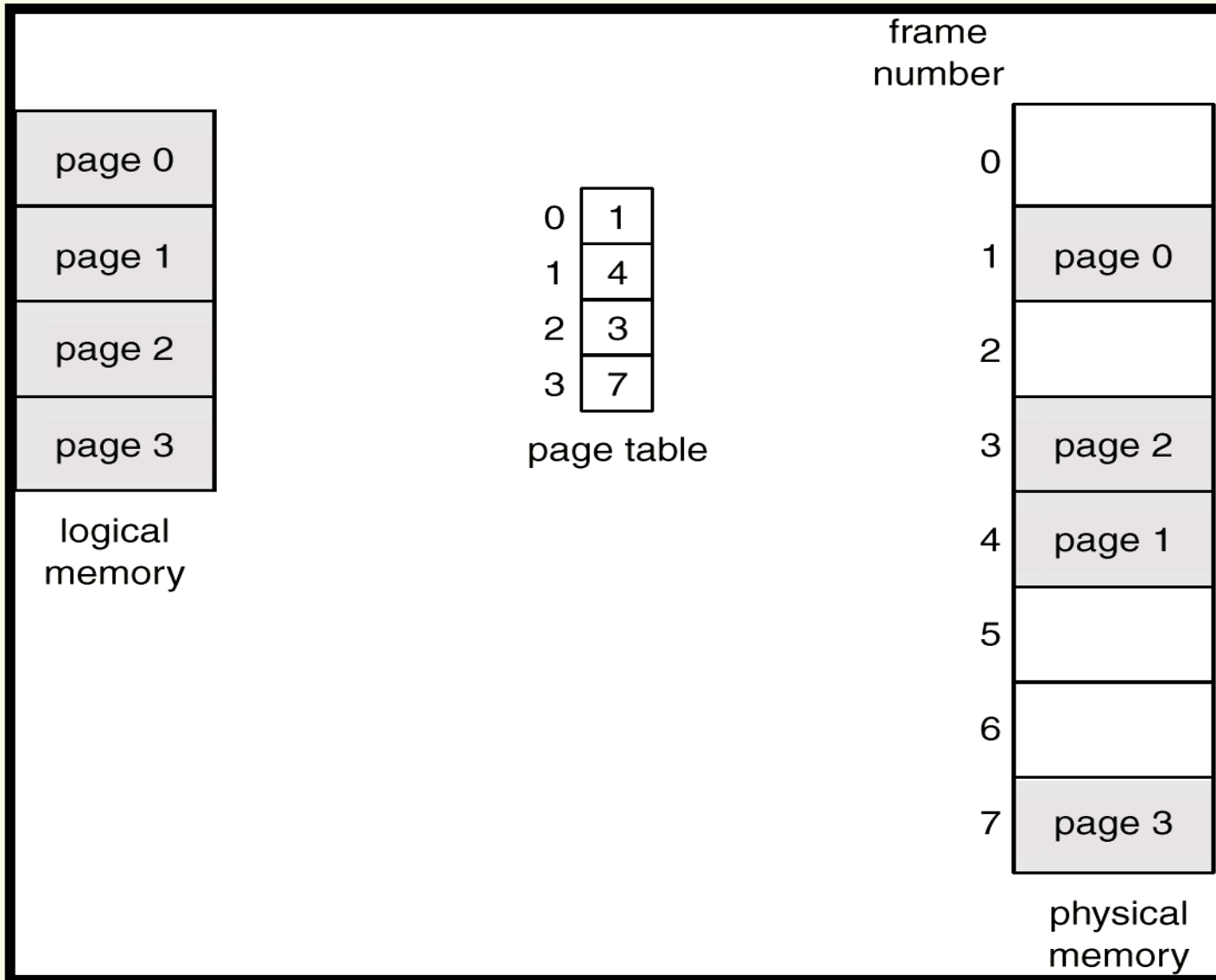
$$f(a) = f(p, b) = p' + b$$

przy czym adres a , numer strony p oraz numer bajtu b są powiązane z rozmiarem strony R w następujący sposób:

p = część całkowita ilorazu a/R

b = reszta z dzielenia a/R

Stronicowanie - przykład



- Liczba ramek stron (rozmiar rzeczywistej pamięci) przydzielonych dla procesu bywa zazwyczaj mniejsza niż liczba stron, których on rzeczywiście używa.
- Jest bardzo prawdopodobne, że adresy w programie mogą się odnosić do strony, której w danej chwili nie ma w pamięci głównej.
- Nastąpi wówczas przerwanie – [mechanizm stronicowania](#) zainicjuje przesyłanie brakującej strony z pamięci pomocniczej do głównej.
- Zostaje też odpowiednio uaktualniona tablica stron.
- Do chwili zakończenia przesyłania strony bieżący proces nie będzie się mógł wykonywać.

- Adres strony w pamięci pomocniczej – przechowywany w odrębnej tablicy albo w tablicy stron.
- Jeśli w tablicy stron - w każdym elemencie tablicy stron - bit "obecności"
- Bit obecności wskazuje, czy dana strona znajduje się w pamięci głównej czy pole adresu należy interpretować jako adres ramki strony, czy też jako adres w pamięci pomocniczej.
- Jeśli w chwili pojawienia się błędu braku strony nie ma żadnej pustej ramki strony - trzeba jakąś inną stronę przesłać do pamięci pomocniczej, aby zrobić miejsce dla brakującej strony.
- Wyboru strony, która ma być w tym celu odesłana -

algorytmu wymiany stron (page turning algorithm)

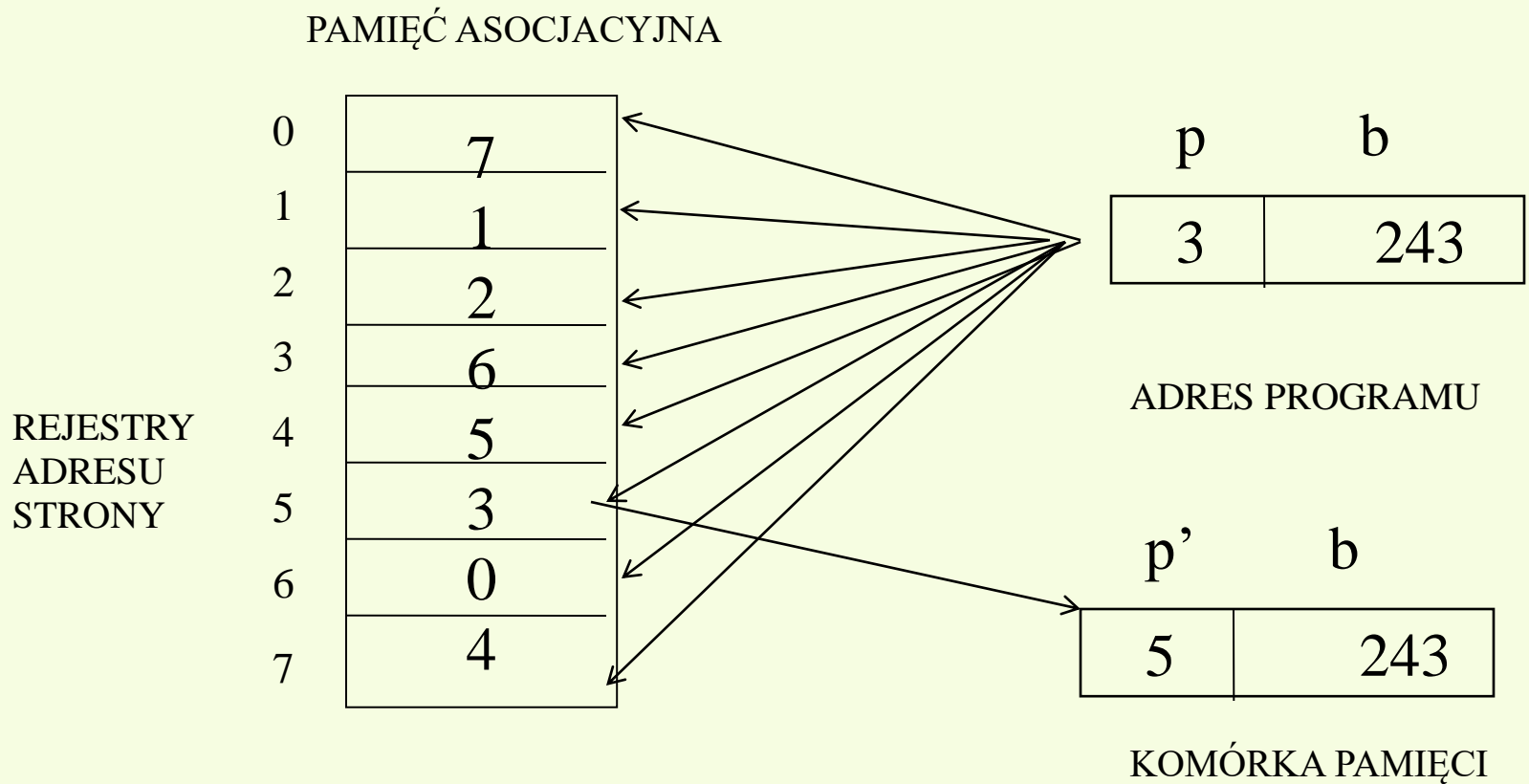
- Jeśli tablica stron w pamięci operacyjnej -- czas każdego odniesienia do pamięci jest dwukrotnie dłuższy
- Można tego uniknąć - tablica stron w zestawie szybkich rejestrów
- Konieczność użycia dużej liczby rejestrów.
- Inne rozwiązanie

pamięć asocjacyjną

zawiera ona mały zbiór rejestrów adresu strony
(page address register)

przechowujących numery stron aktywnych

Odwzorowanie adresu przy użyciu pamięci asocjacyjnej



Przykład

strona składa się z 1000 bajtów

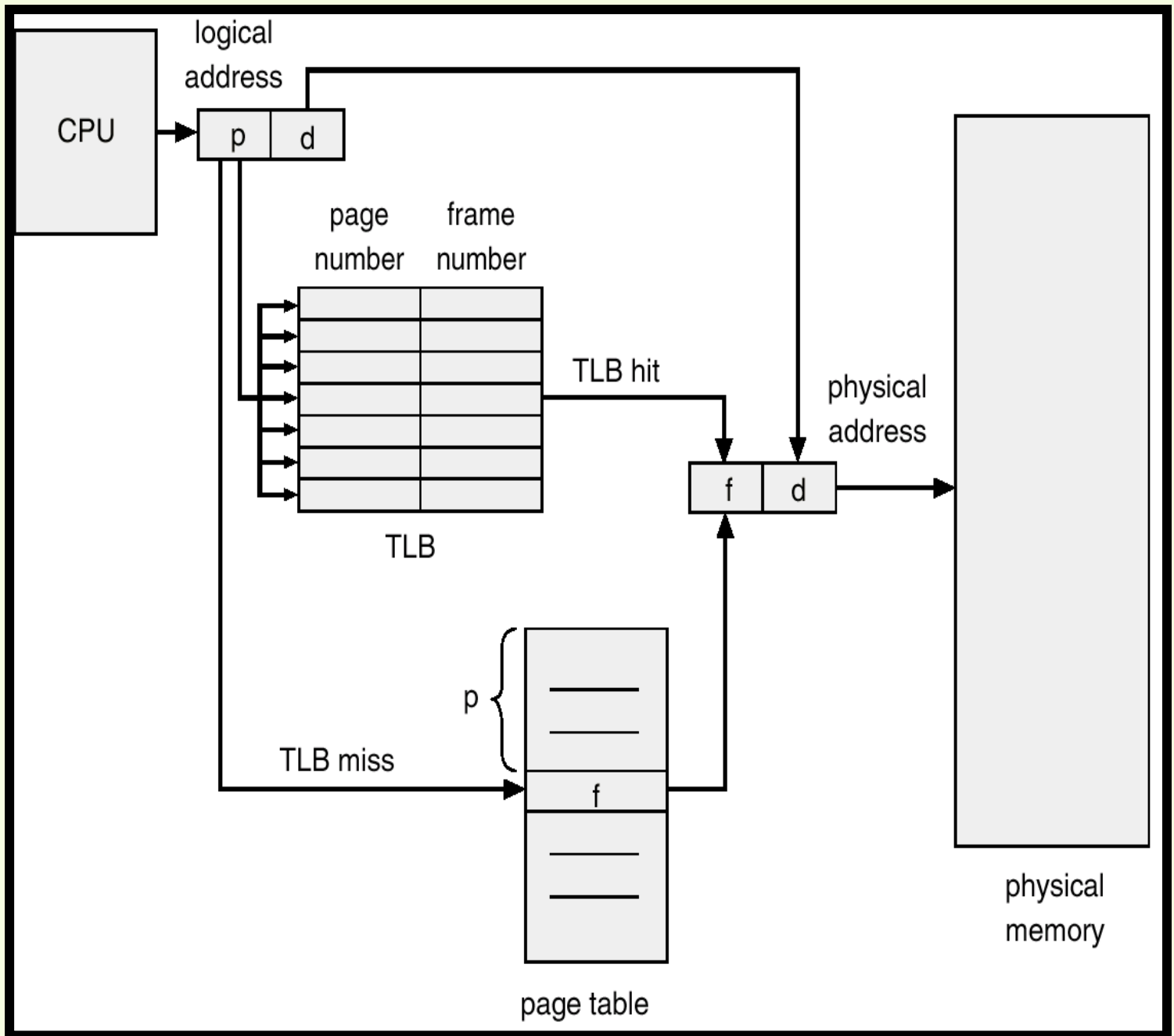
- występujący w programie adres 3243
składa się z dwóch części:
 - numeru strony 3
 - numeru bajtu 243
- Numer strony jest porównywany
z zawartością wszystkich rejestrów adresu strony
– adres rzeczywisty: 5243

Pamięć asocjacyjna

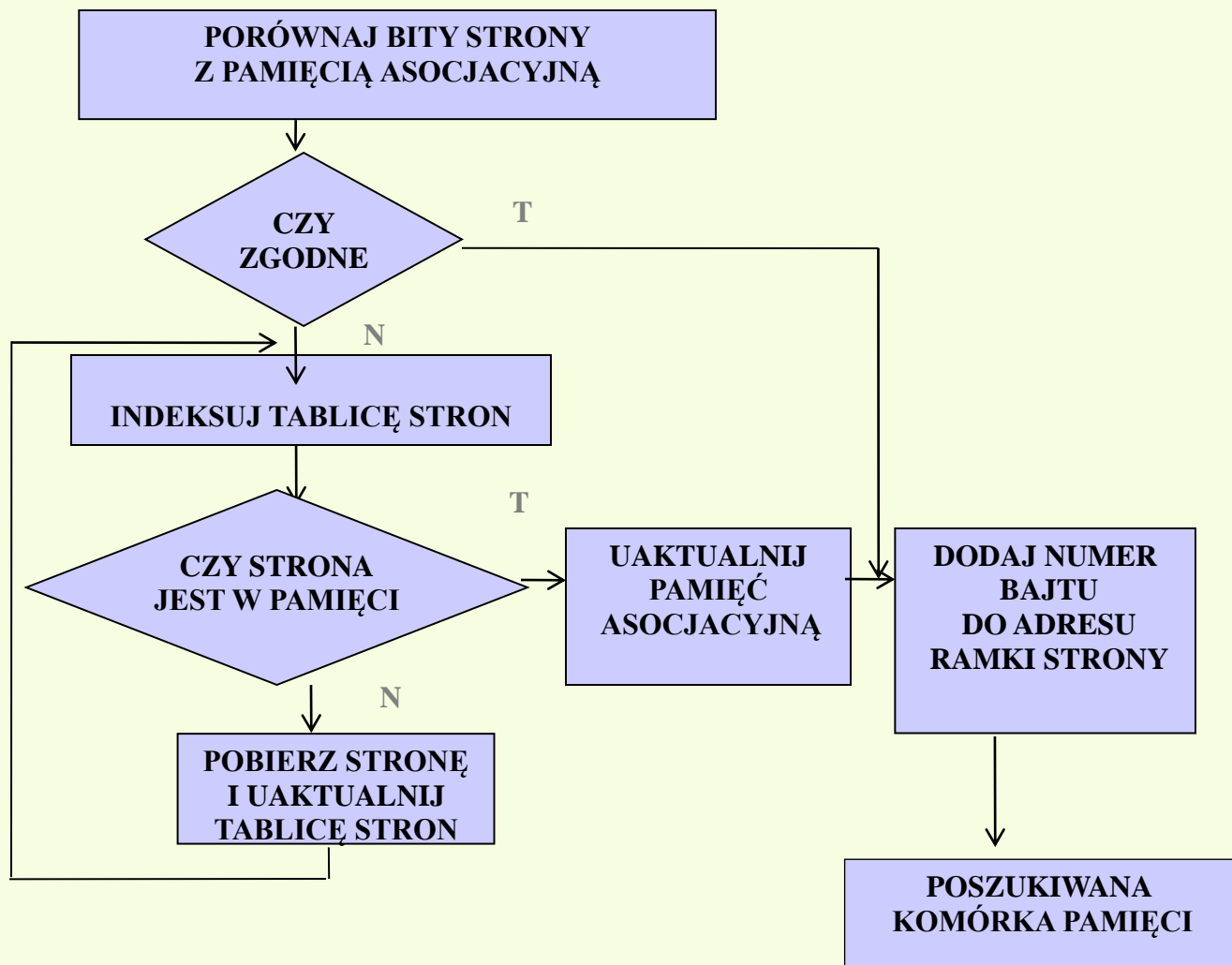
- zmniejsza koszty o rząd wielkości.
- Liczba pozycji = liczba ramek stron w pamięci głównej.

kompromis:

- Pełna tablica stron dla każdego procesu - w pamięci głównej.
- mała pamięć asocjacyjna - kilka stron ostatnio aktywnych procesów
(bufory translacji adresów stron -TLBs
Translation Look-Aside Buffers 8-2048 pozycji)
- do wyznaczenia numeru ramki strony - dodatkowe pole w pam. asocjacyjnej



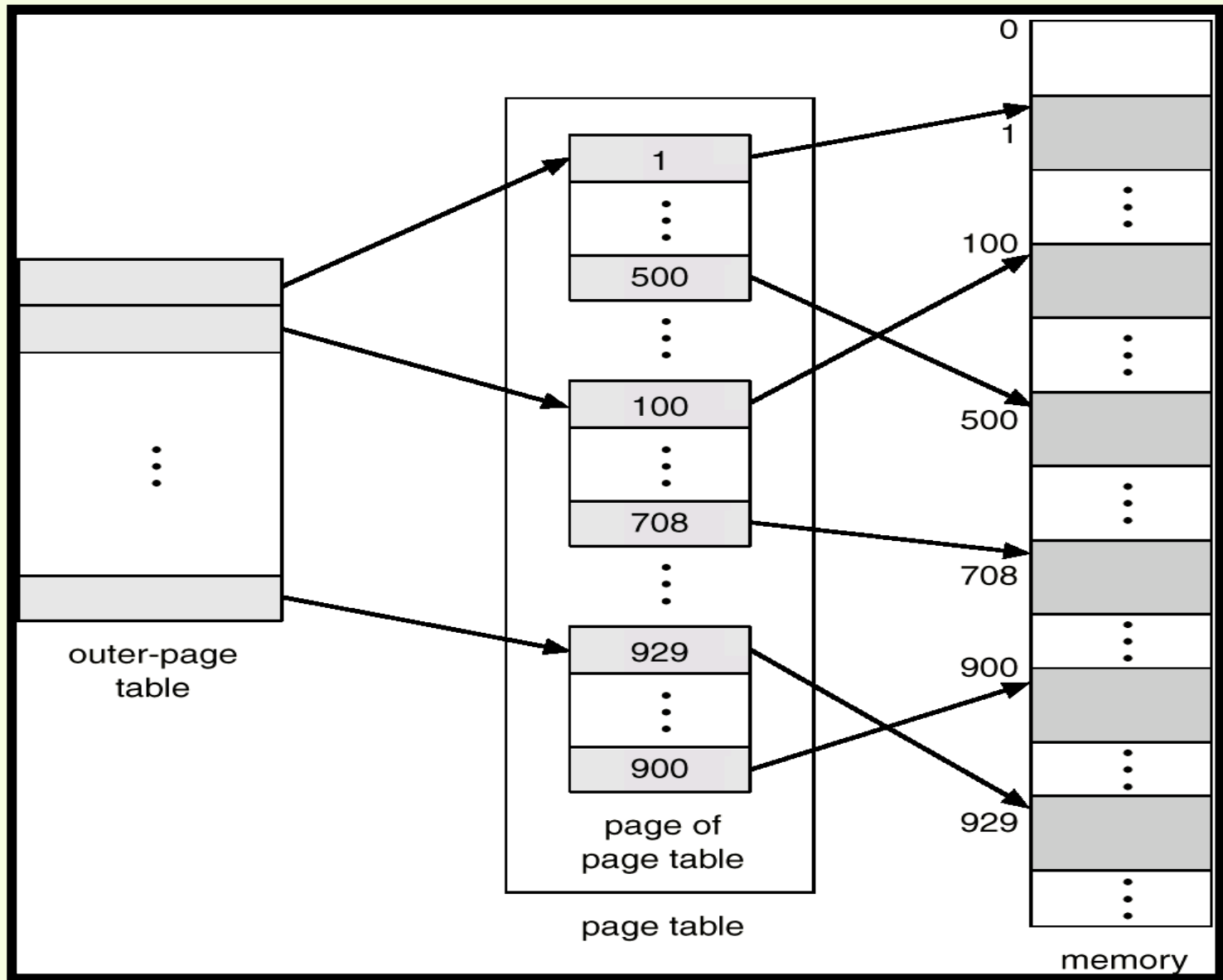
Operacja odwzorowania adresu przy użyciu stronicowania i małej pamięci asocjacyjnej



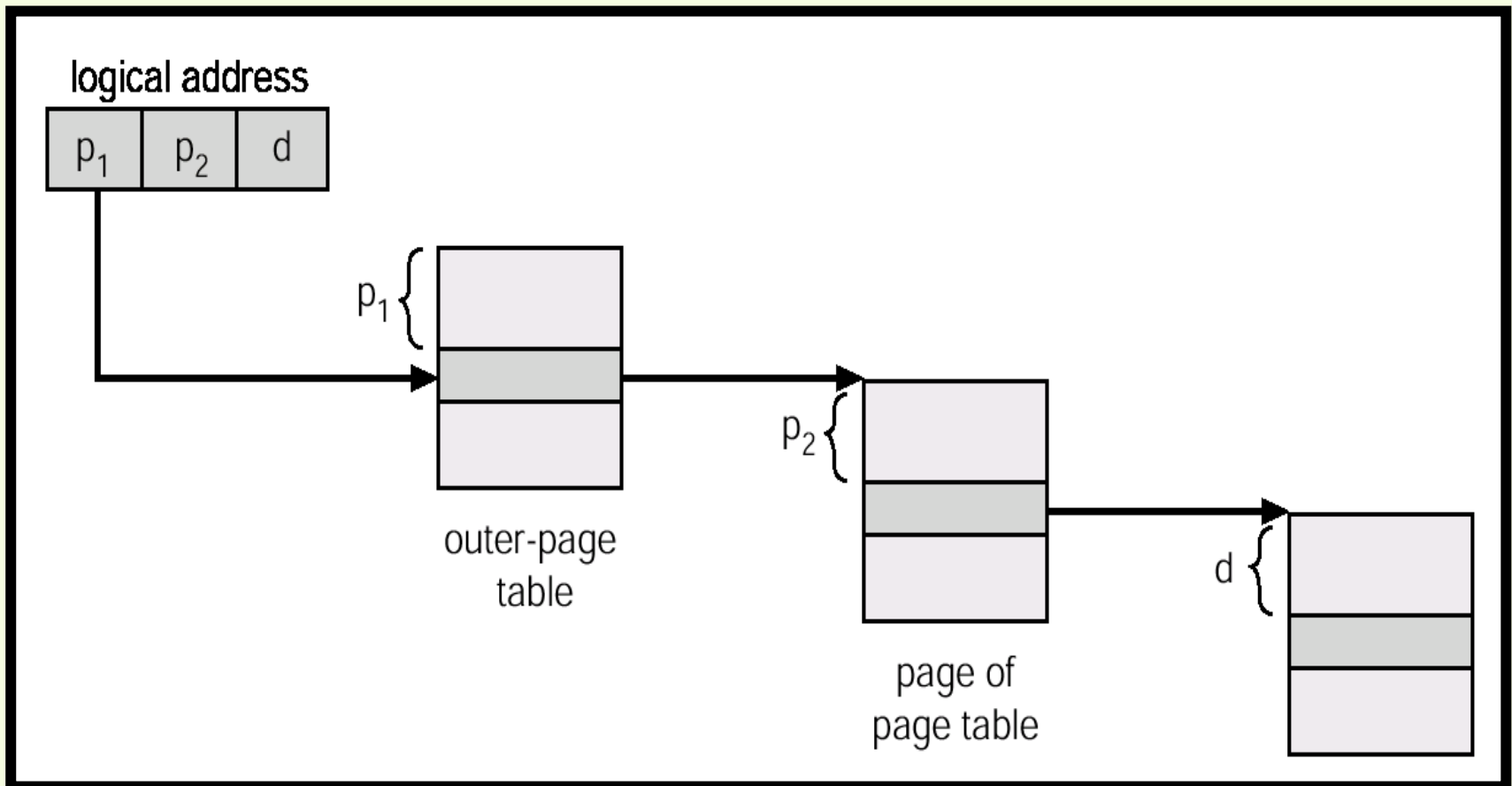
Stronicowanie wielopoziomowe

- Przestrzeń adresowa – 32-bitowa
- Rozmiar strony – 4kB - 2^{12} B
- Rozmiar tablicy stron – do 10^6 wpisów po 4B
 $2^{32}/2^{12}=2^{20}$ - wpisów po 4B = 4MB –tab. stron 1 procesu
- Stronicowanie wielopoziomowe (10 + 10 + 12)

Stronicowanie wielopoziomowe



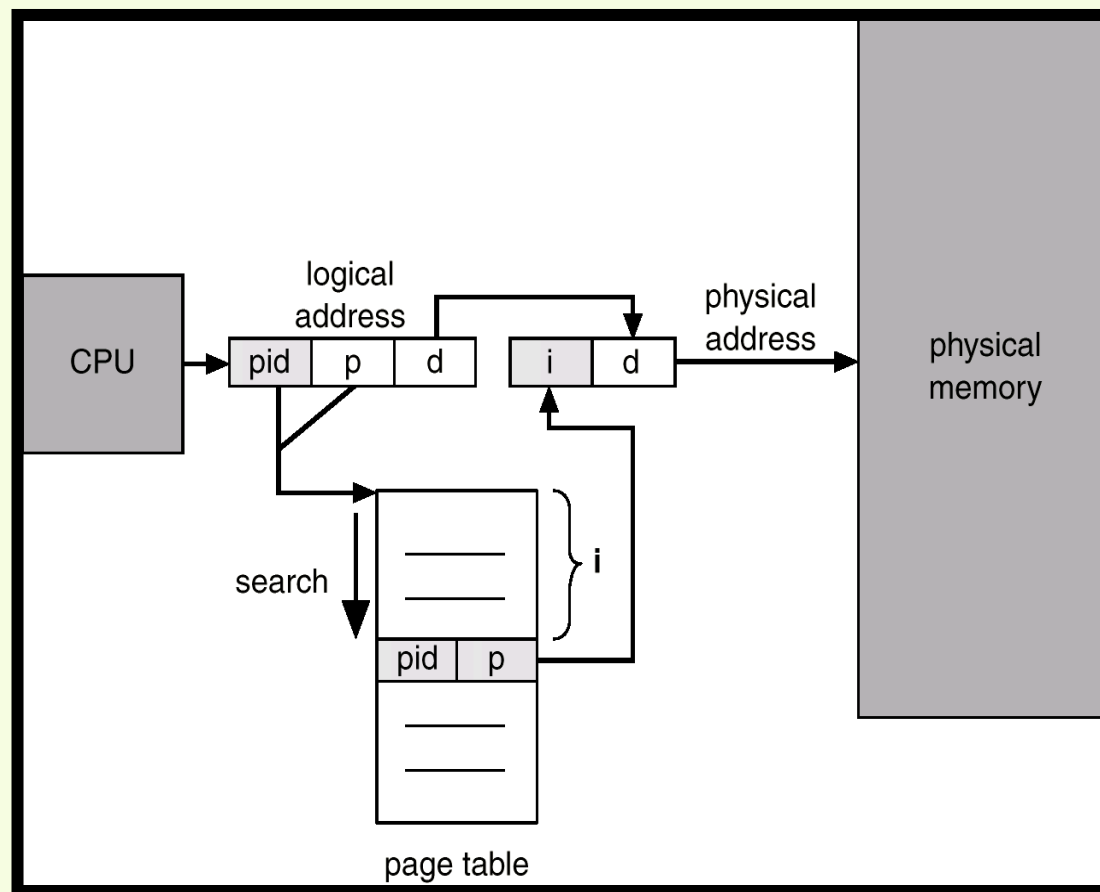
Stronicowanie wielopoziomowe – tłumaczenie adresu



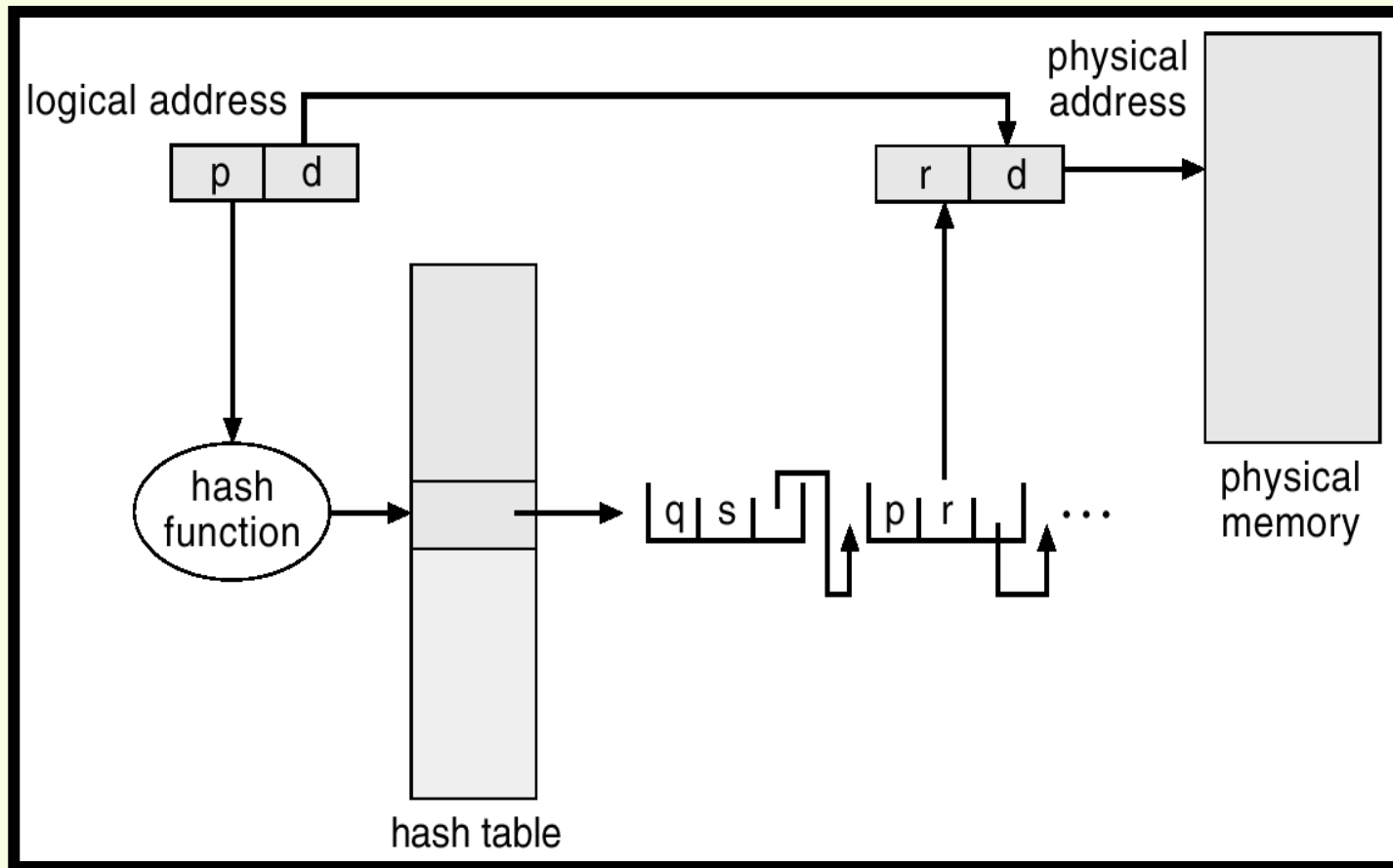
Stronicowanie wielopoziomowe

- SPARC (32 bitowy adres –
3-poziomowe stronicowanie)
- Motorola 68030 – 4-poziomowe stronicowanie
- Każdy poziom – osobna tablica w PAO;
zastosowanie pamięci podręcznej

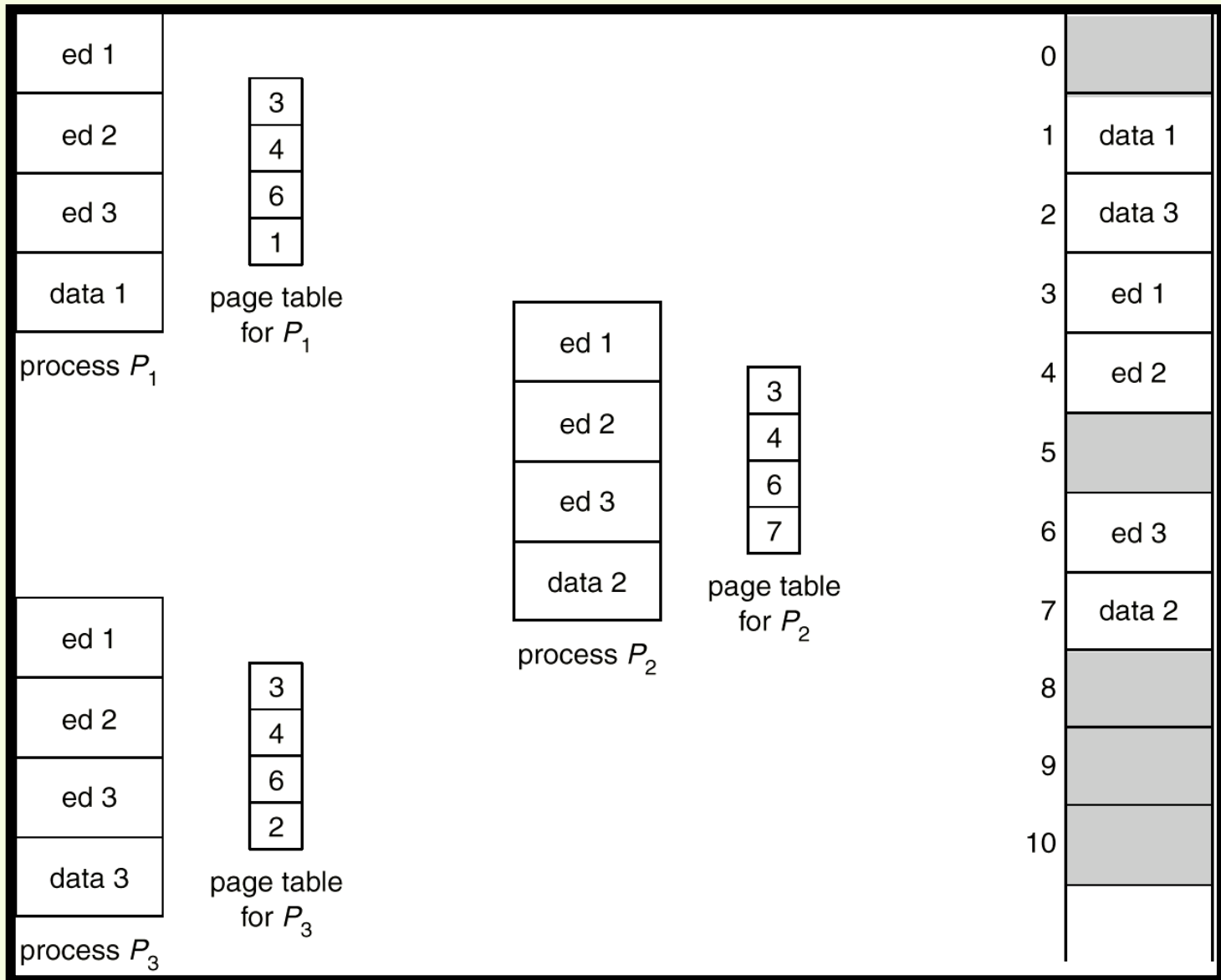
Odwrócona tablica stron



Tablica haszowania

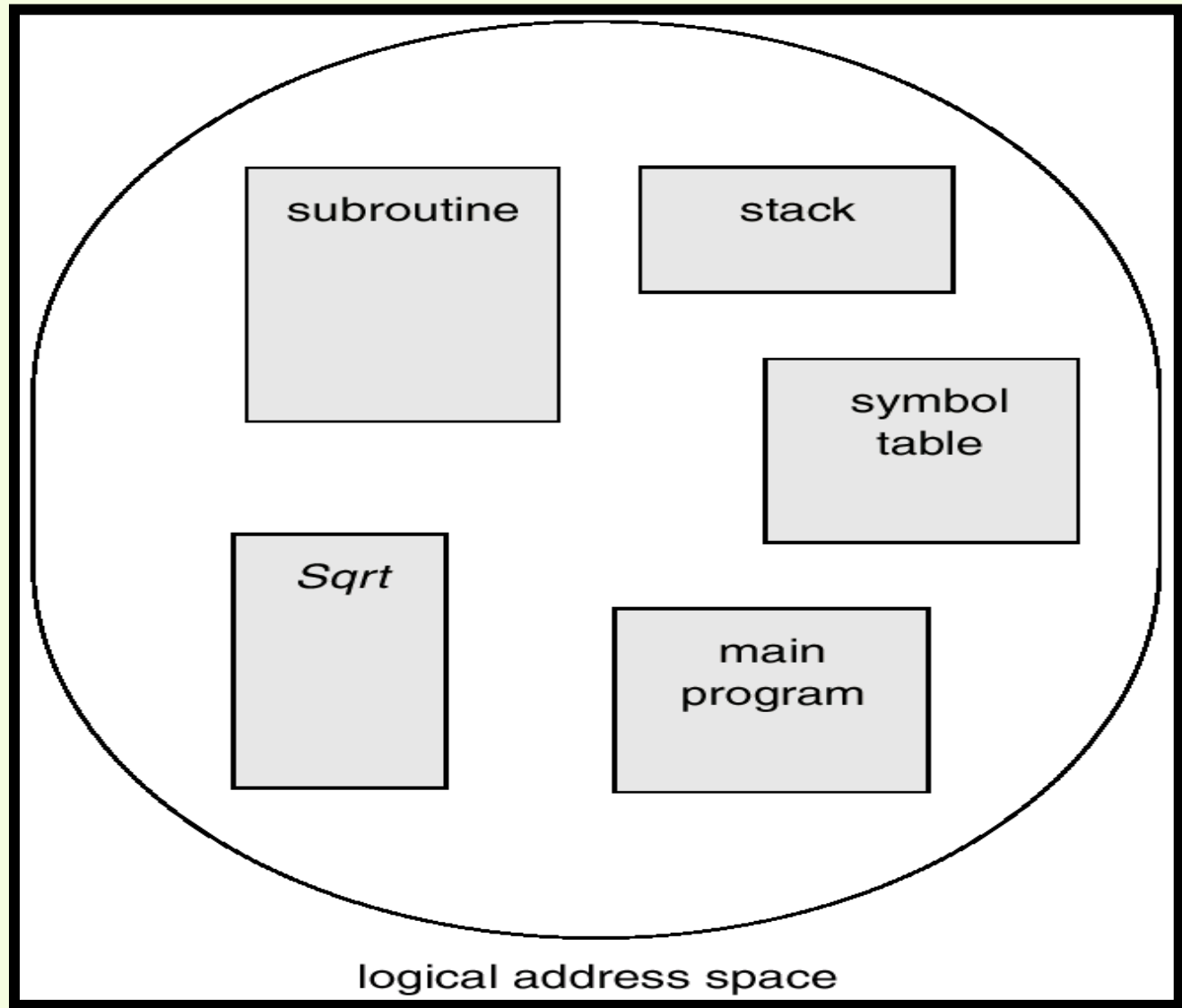


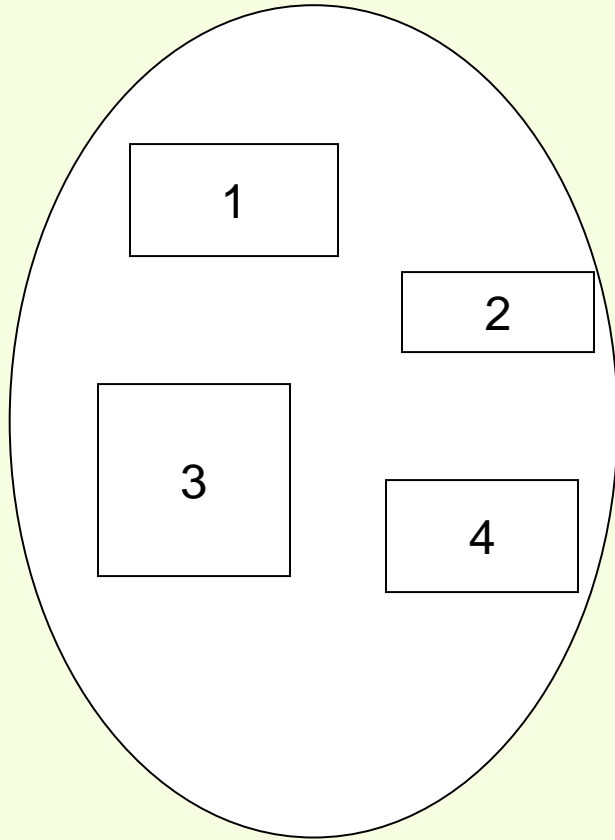
Dzielone strony



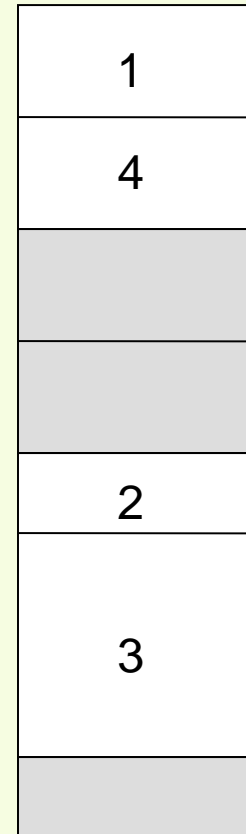
segmentacja

segmentacja





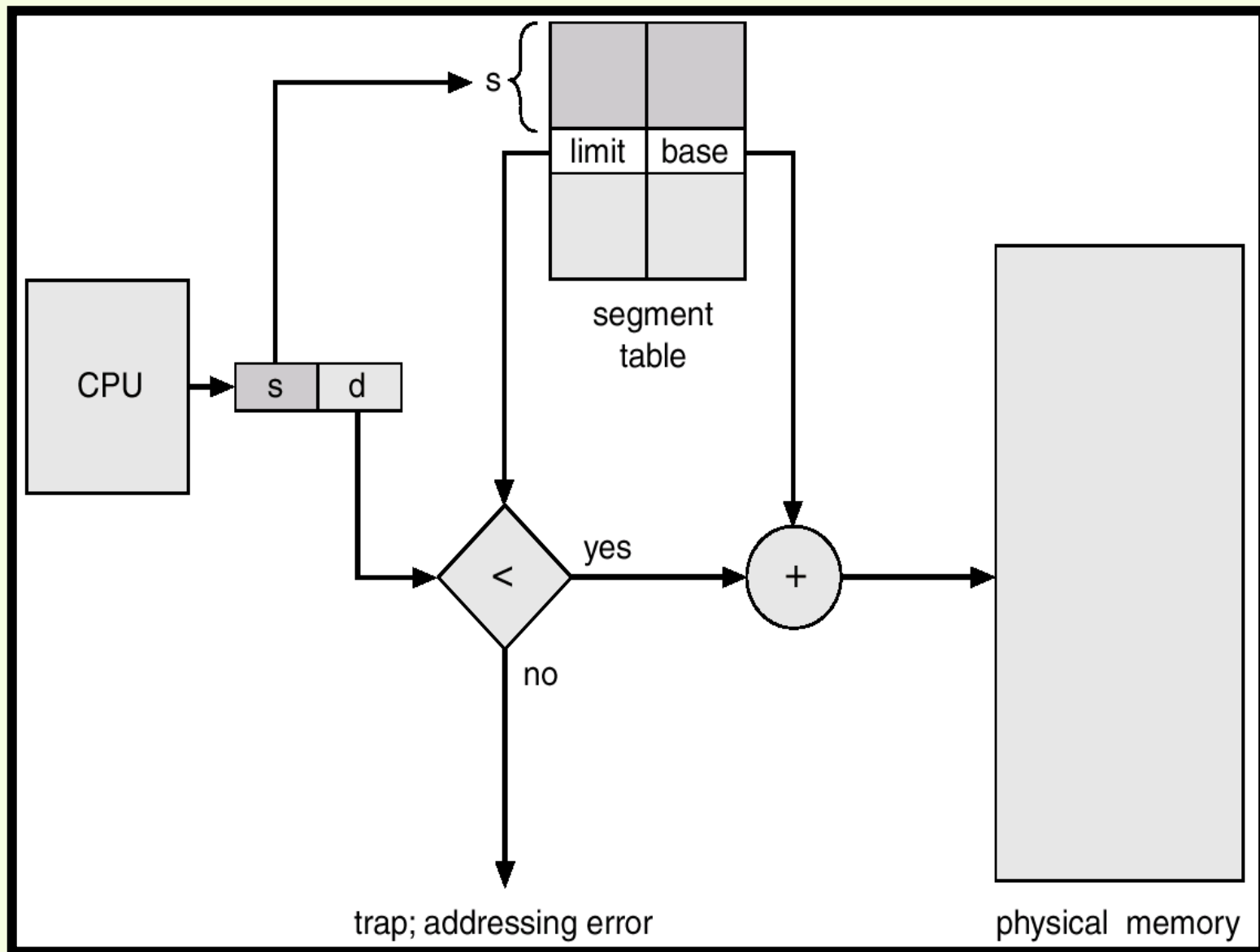
user space



physical memory
space

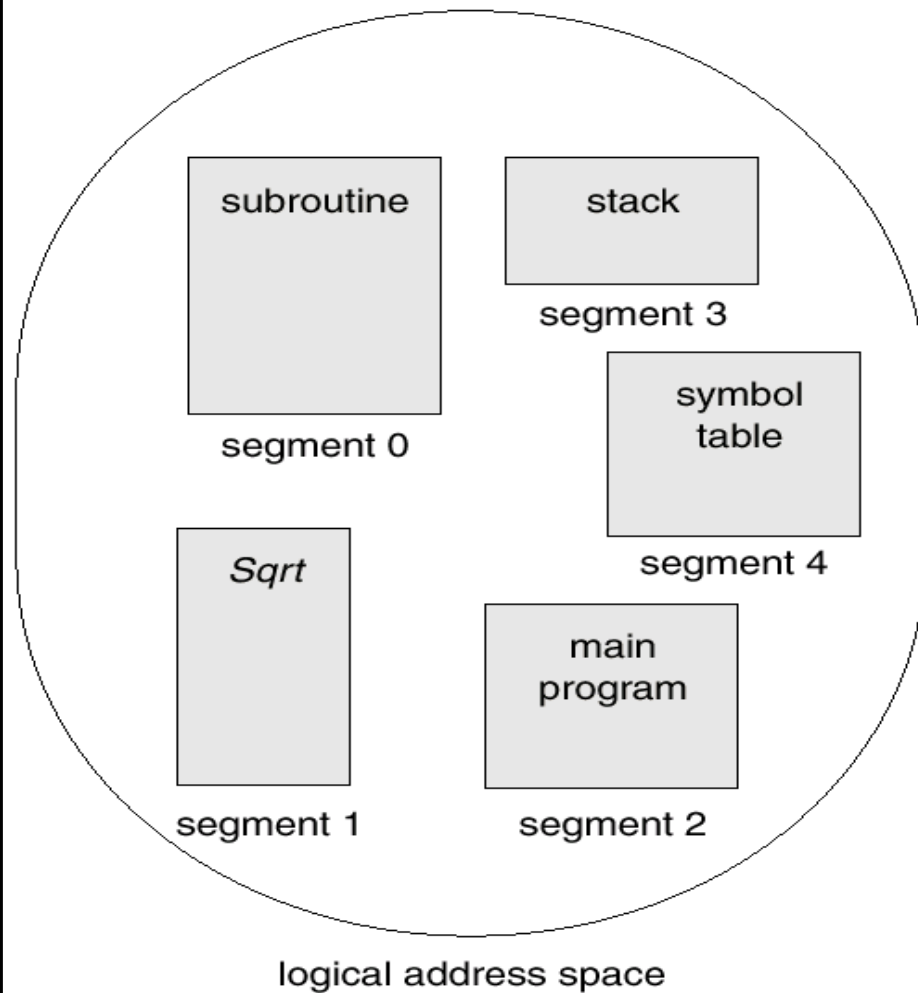
Segmentacja pamięci

- Segmentacja - odzwierciedla logiczny podział informacji na program i dane.
- Przestrzeń adresów dzieli się na segmenty
 - segmenty odpowiadają poszczególnym procedurom....
- Kilka par rejestrów bazowych i granicznych dla każdego procesu – w przestrzeni adresów kilka odrębnych obszarów.
- Wada - mała liczba segmentów oraz konieczność określania z góry przeznaczenia poszczególnych
- Przestrzeń adresów - dwuwymiarowa
 - adresy w programie: nazwa segmentu i adres wewnątrz segmentu.



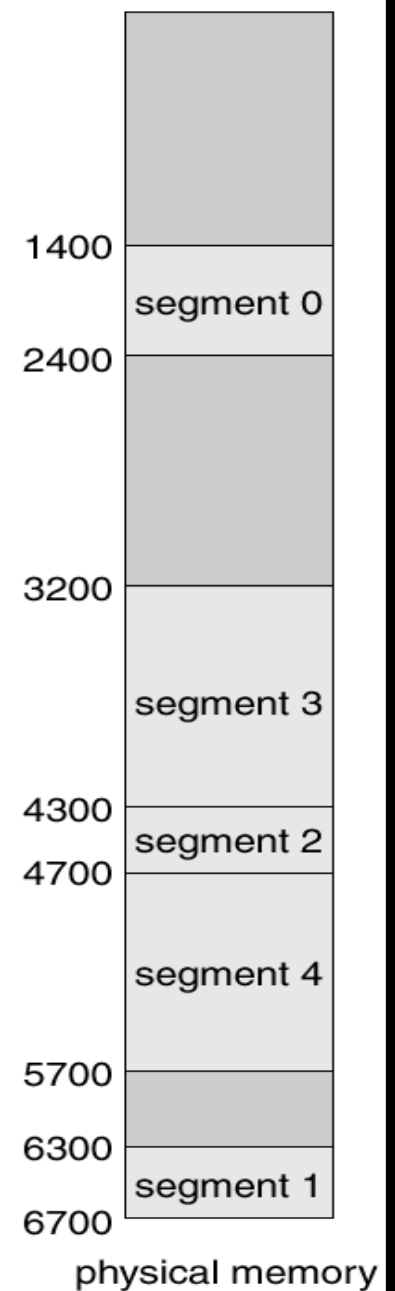
Segmentacja pamięci

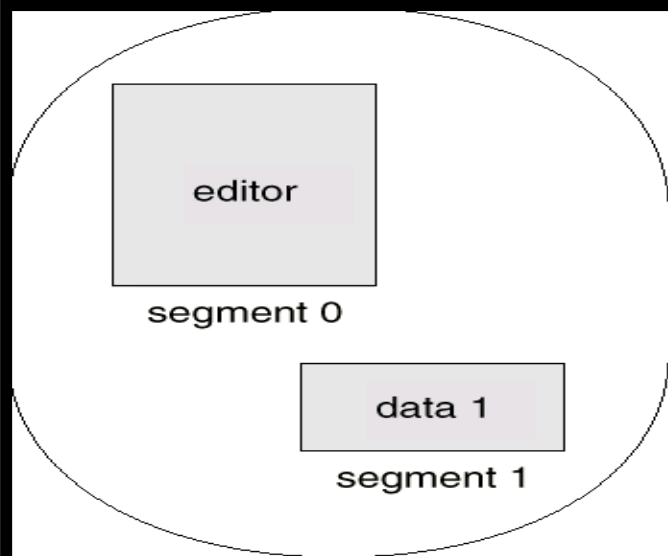
- Adres w programie: (s, a)
 - s - numer segmentu,
 - a - adres wewnątrz tego segmentu
- dla każdego procesu - tablica segmentów (deskryptory):
 - adres bazy
 - długość segmentu s danego procesu.



	limit	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

segment table

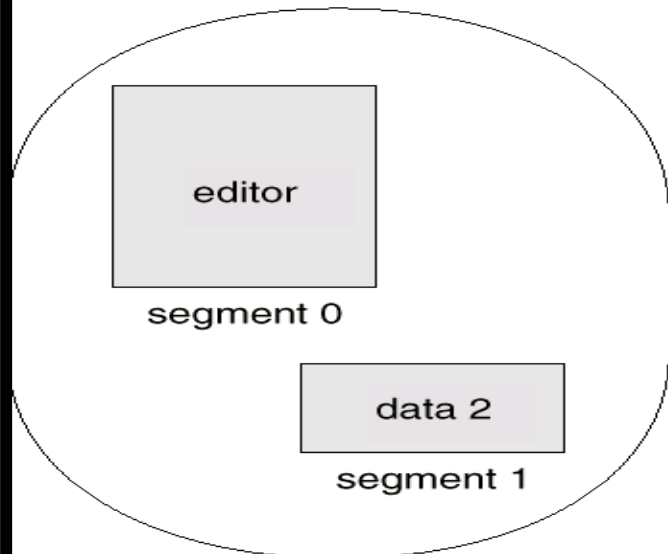




logical memory
process P_1

	limit	base
0	25286	43062
1	4425	68348

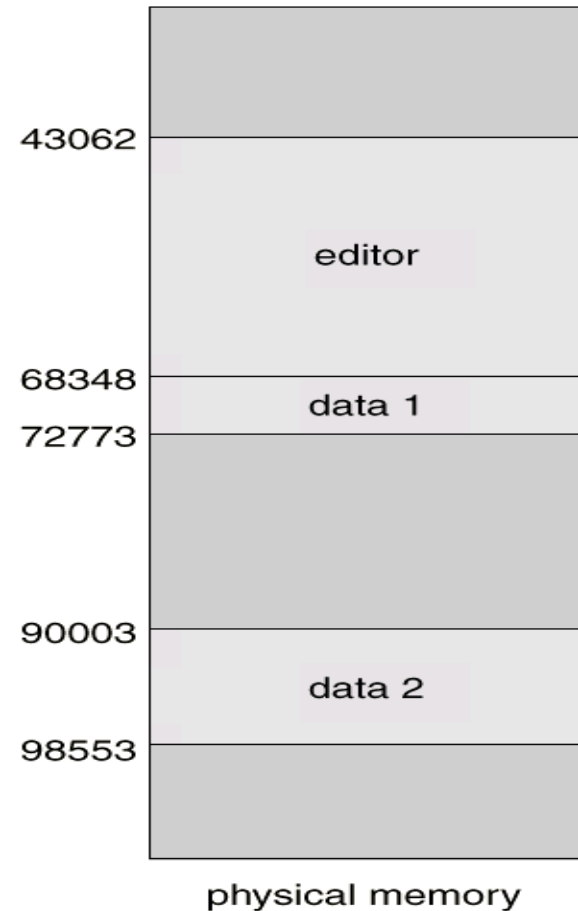
segment table
process P_1



logical memory
process P_2

	limit	base
0	25286	43062
1	8850	90003

segment table
process P_2



stronicowanie a segmentacja pamięci

- Cel segmentacji –
logiczny podział przestrzeni adresów
- Cel stronicowania –
fizyczny podział pamięci
- Strony mają ustalony rozmiar
- Rozmiar segmentów może być dowolny

stronicowanie a segmentacja pamięci

stronicowanie

- Podział adresu na numery strony i bajtu
 - wykonywany sprzętowo
- przekroczenie zakresu dla numeru bajtu
 - automatyczne zwiększenie numeru strony

segmentacja

- Podział adresu programu na numery segmentu i bajtu
 - logiczny
- przekroczenie zakresu dla numeru bajtu
 - sygnalizacja przekroczenia zakresu pamięci

Segmentacja pamięci

- stronicowanie segmentów, albo wymiana całych segmentów
- Stronicowanie segmentów –
każdy segment składa się ze stron i ma własną tablicę stron

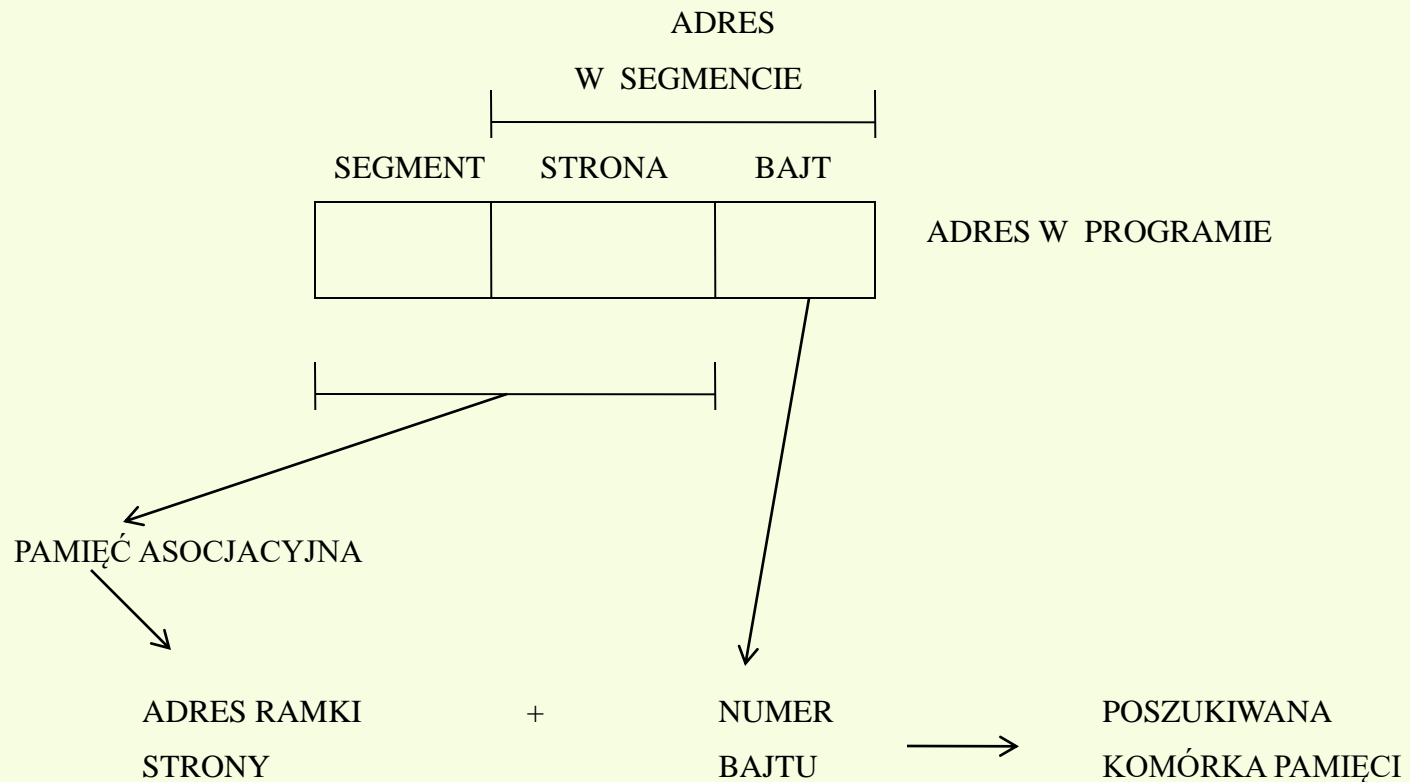
SEGMENTACJA + STRONICOWANIE

- Wydziel w programie adres (s, a)
- Użyj s do indeksowania tablicy segmentów
- Jeżeli element s tablicy segmentów jest pusty utwórz nową tablicę stron,
w przeciwnym przypadku wydziel adres tablicy stron
- Wydziel w adresie a numer p strony i numer b bajtu
- Użyj p do indeksowania tablicy stron
- Jeżeli element p tablicy stron jest pusty pobierz stronę
z pamięci pomocniczej, w przeciwnym przypadku
wydziel adres p' ramki strony
- Dodaj p' do numeru b bajtu -- adres komórki pamięci

Tworzenie adresu przy pomocy pamięci asocjacyjnej

- Użyj pamięci asocjacyjnej - są tam numery **segmentów** i **stron** – ostatnio używane
- Porównaj bity segmentu i strony adresu z zawartością pamięci asocjacyjnej.
- Jeśli zgodne - dodaj numeru bajtu do adresu ramki strony --- adres komórki pamięci.

odzworowanie adresu przy użyciu segmentów stronicowanych i pamięci asocjacyjnej



Stronicowanie + segmentacja

- Každy segment posiada własną tablice stron:
 - System MULTICS (Honeywell 6180)
 - IBM 370
 - Segmentacja i stronicowanie - niezależne
 - Intel \geq 80386
 - Segmentacja + stronicowanie mogą być stosowane łącznie lub rozdzielnie; niezależnie od siebie
 - 1 bit rejestru sterującego procesora – stronicowanie on(1)/off (0)
 - jeśli 0 – adres liniowy = adres fizyczny
 - Segmenty – w pojedynczej przestrzeni adresowej realizowanej za pomocą stronicowania
- 286 – segmentacja
- 386 segmentacja + stronicowanie

PAMIĘĆ WIRTUALNA

- technika umożliwiająca wykonanie procesów, które nie są w całości przechowywane w PAO
- program może być większy niż pamięć fizyczna
- pozwala utworzyć abstrakcyjną pamięć główną
- oddziela pamięć logiczną od pamięci fizycznej
- ułatwia proces programowania
- implementacja pamięci wirtualnej może obniżyć wydajność

Wykonywane rozkazy muszą być w PAO

- cała logiczna przestrzeń adresowa w pamięci fizycznej
- nakładki; ładowanie dynamiczne
- pamięć wirtualna

W wielu przypadkach cały program nie jest potrzebny (np.):

- obsługa rzadko pojawiających się błędów
- nadmiarowe deklaracje tablic i wykazów
- pewne możliwości programów są b. rzadko wykorzystywane

Zalety częściowego zaalokowania programów w PAO

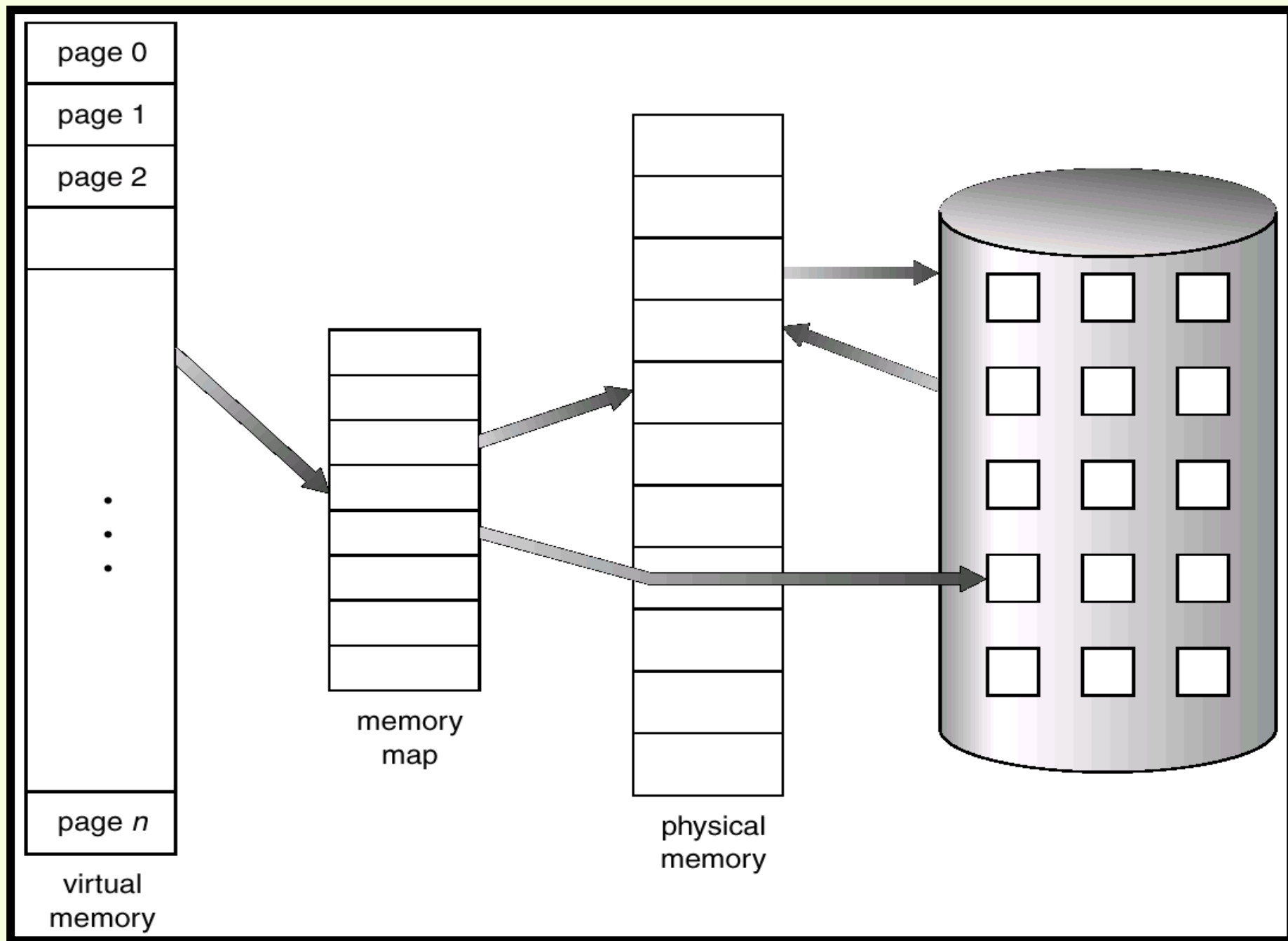
- wygoda programistów (brak ograniczeń pamięci)
- zwiększenie stopnia wieloprogramowości,
a zatem wykorzystania procesora, przepustowości
- szybsze wykonanie programu użytkownika (mniej operacji we/wy)

IMPLEMENTACJA PAMIĘCI WIRTUALNEJ

- stronicowanie na żądanie

procedura leniwej wymiany (lazy swapper) (stronicująca)

- segmentacja na żądanie



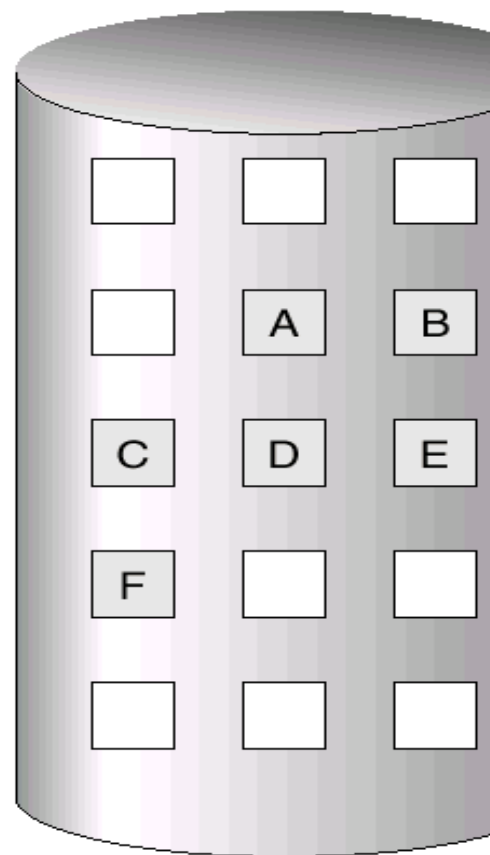
0	A
1	B
2	C
3	D
4	E
5	F
6	G
7	H

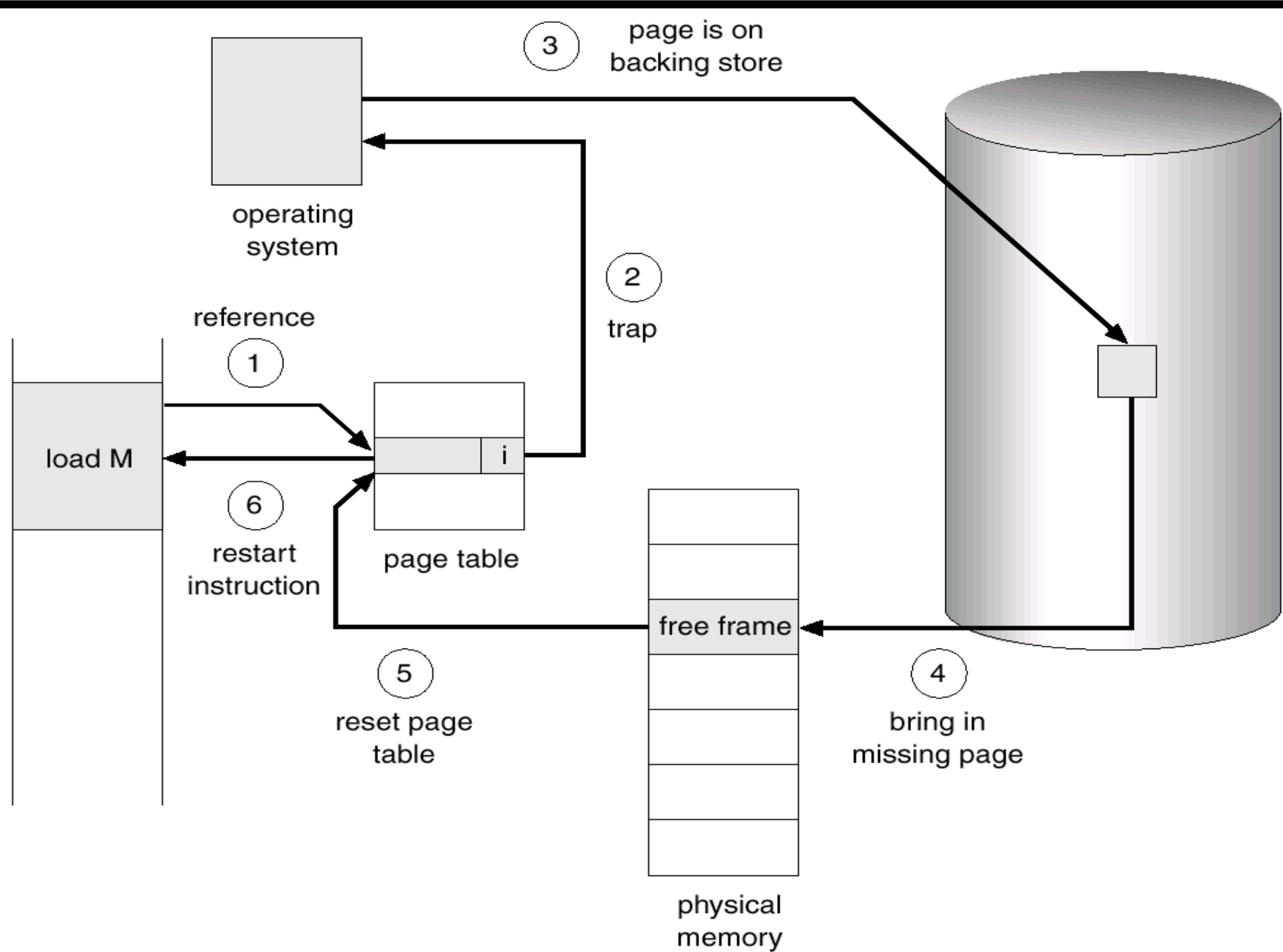
The diagram shows a table representing a valid-invalid bit frame. The table has 8 rows, indexed 0 to 7. The first column is labeled 'frame' and the second column is labeled 'valid-invalid bit'. The entries in the 'valid-invalid bit' column are: 0: 4, 1: (empty), 2: 6, 3: (empty), 4: (empty), 5: 9, 6: (empty), 7: (empty). The 'valid-invalid bit' column contains either a number or is empty, indicating the validity of the frame.

frame	valid-invalid bit
0	4
1	
2	6
3	
4	
5	9
6	
7	

page table

0	
1	
2	
3	
4	A
5	
6	C
7	
8	
9	F
10	
11	
12	
13	
14	
15	





Stronicowanie na żądanie

wyposażenie sprzętowe:

(tablica stron, pamięć pomocnicza) + oprogramowanie
po wystąpieniu braku strony - wznowienie rozkazu

dodaj(A,B,C)

- pobranie i zdekodowanie rozkazu dodaj
- pobranie A
- pobranie B
- dodanie A i B
- zapamiętanie sumy w C (C jest na stronie, której nie ma w PAO)
 - sprowadzenie strony
 - powtórne wykonanie rozkazu (pobranie)

problem - np. przesyłanie grupy bajtów
zachodzących na siebie,
leżących na granicy stron –
brak strony może wystąpić po częściowym przesłaniu
(dane w bloku źródłowym mogą być już zmienione)

rozwiązania:

- mikroprogramowe: obliczenie obu końców obu bloków;
jeśli brak strony - obsługiwany przed zmianami
- użycie rejestrów do chwilowego przetrzymywania
wartości przesyłanych pól;
w przypadku pułapki - odtworzenie poprzednich wartości

SPRAWNOŚĆ STRONICOWANIA NA ŻĄDANIE

- efektywny czas dostępu *ecd*

$$cd = 10 - 200 \text{ ns}; (100)$$

cobs - czas obsługi braku strony

(obsługa przerwania, czytanie strony, wznowienie procesu)

p - prawdopodobieństwo braku strony

$$ecd = (1-p) \cdot cd + p \cdot cobs$$

$$cobs = 25 \text{ ms (bez kolejki)}$$

$$ecd = 100 + 24\,999\,900 \cdot p \text{ [ns]}$$

dla $p = 0.001$ - $ecd \approx 25000 \text{ ns}$ - **250-krotne spowolnienie komputera**

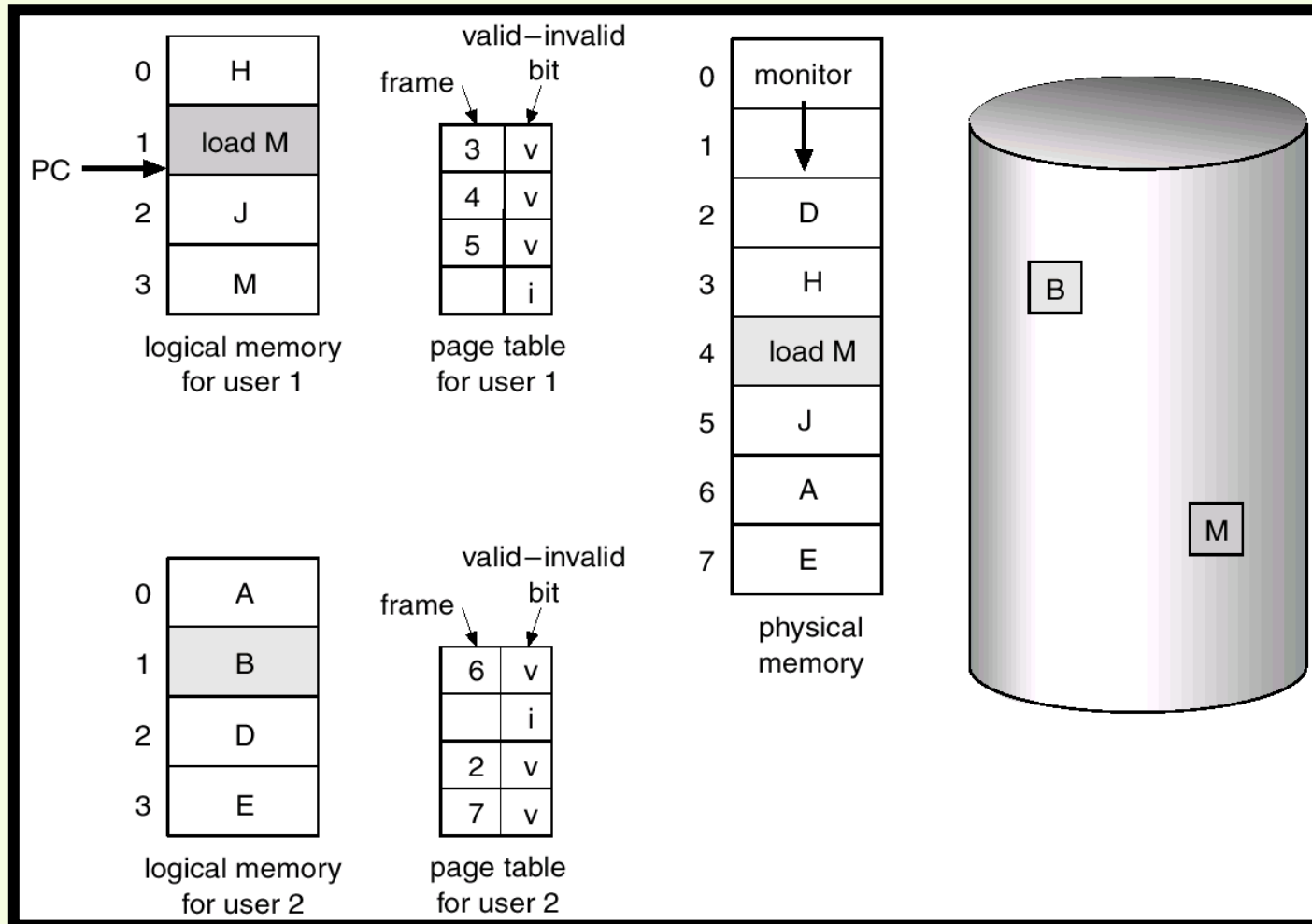
aby pogorszenie $< 10\%$ - $p < 0.00000004$

(1 brak strony na 2 500 000 odwołań do pamięci)

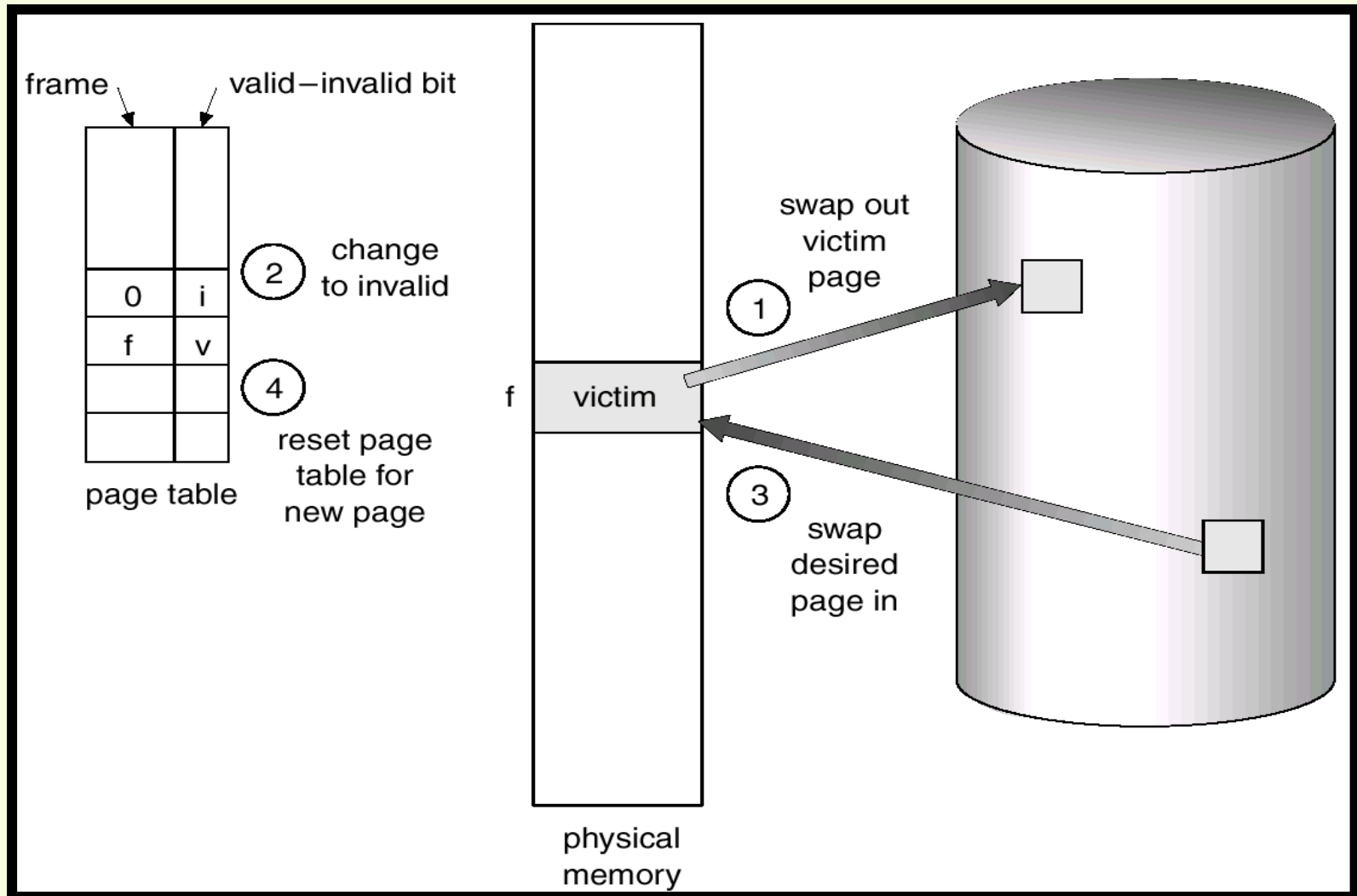
ZASTĘPOWANIE STRON

powiększenie stopnia wieloprogramowości - nadprzydział pamięci
(brak wolnych ramek)

- zakończenie procesu użytkownika
- wymiana - zmniejszenie poziomu wieloprogramowości
- zastępowanie stron



- wybór „ramki ofiary”
- dwukrotne przesyłanie stron (na dysk i z dysku) - wydłużenie efektywnego czasu dostępu
- zastosowanie *bitu modyfikacji - bitu zabrudzenia*



ALGORYTMY ZASTĘPOWANIA STRON

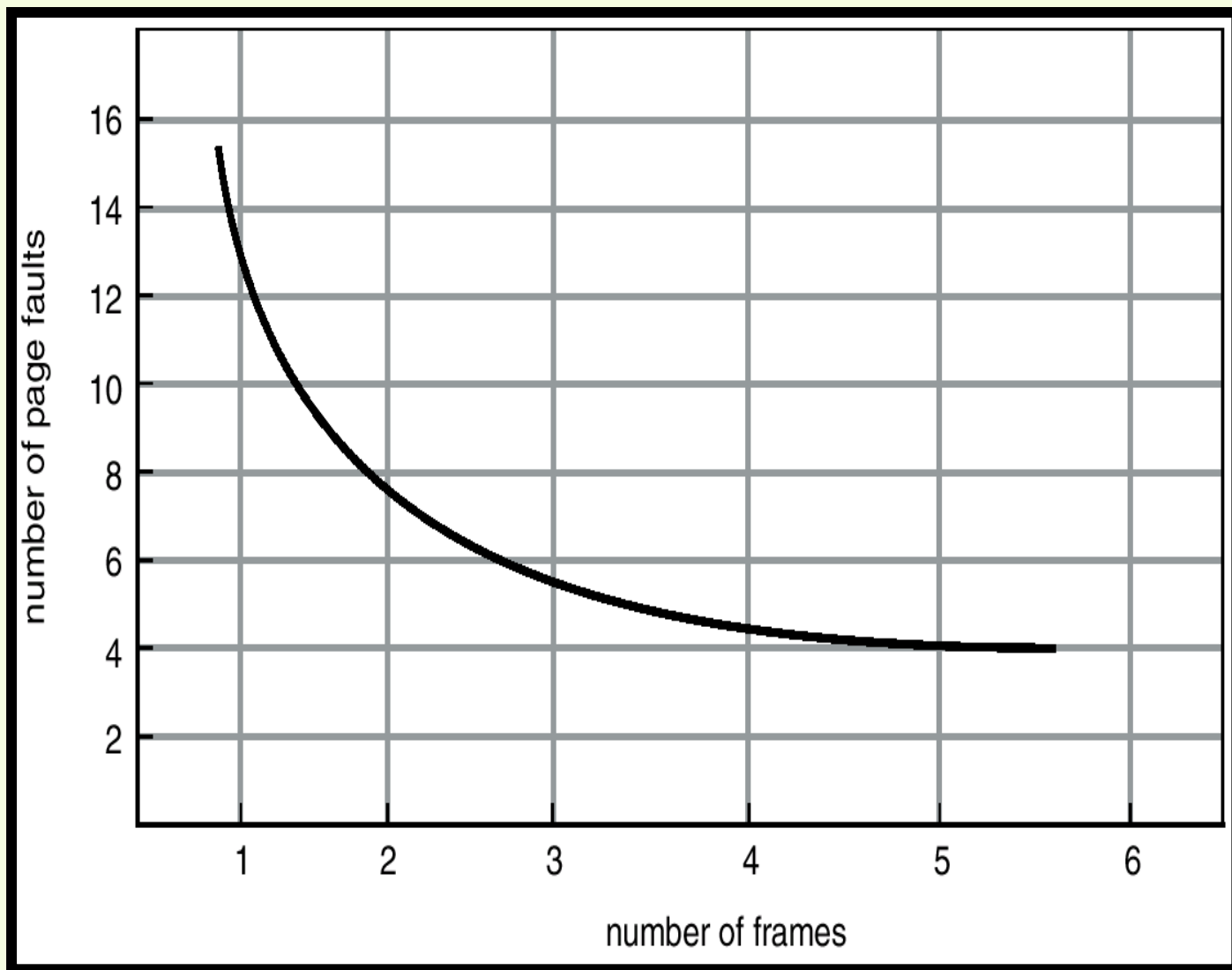
- minimalizacja częstości braków stron
- oceniany na pdst. ciągu odniesień do pamięci
(ciąg adresów zredukowany do numerów kolejnych różnych stron) przy znanej liczbie ramek

np. ciąg adresów przy 100-bitowej stronie:

0100, 0432, 0101, 0612, 0102, 0103, 0104,
0101, 0611, 0102, 0103, 0104, 0101, 0601,
0102, 0103, 0104, 0101, 0609, 0102, 0105

można zredukować do: 1, 4, 1, 6, 1, 6, 1, 6, 1, 6, 1

liczba braków stron maleje wraz ze wzrostem liczby ramek do pewnego minimalnego poziomu



ALGORYTMY ZASTĘPOWANIA STRON

- FIFO
- algorytm optymalny
- LRU
- algorytmy przybliżające LRU
- algorytmy zliczające
- algorytm buforowania stron

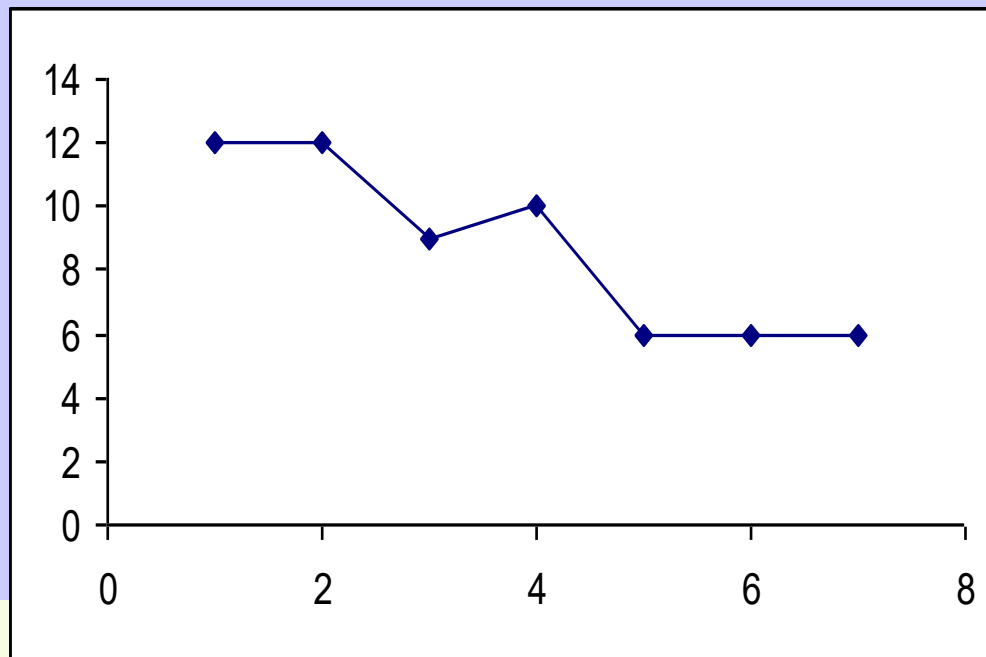
7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

FIFO

- 15 braków stron
- dla ciągu odniesień: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 -

anomalia Belady'ego

wsp.braku stron wzrasta ze wzrostem liczby ramek



FIFO

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2		2	2	4	4	4	0			0	0			7	7	7
	0	0	0		3	3	3	2	2	2			1	1			1	0	0
		1	1		1	0	0	0	3	3			3	2			2	2	1

Algorytm optymalny

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2		2		2			2			2				7		
	0	0	0		0		4			0			0				0		
		1	1		3		3			3			1				1		

Algorytm optymalny

- najniższy współczynnik braku stron (9)
- nigdy nie występuje anomalia Belady'ego
- istnieje - OPT, MIN

Zastąp tę stronę, która najdłużej nie będzie używana

wymaga wiedzy o przyszłej postaci ciągu odniesień

(jak planowanie procesora - SJF)

LRU *Least Recently Used*

- Z każdą stroną kojarzy czas jej ostatniego użycia
- nie występuje anomalia Belady'ego
- (12 błędów)

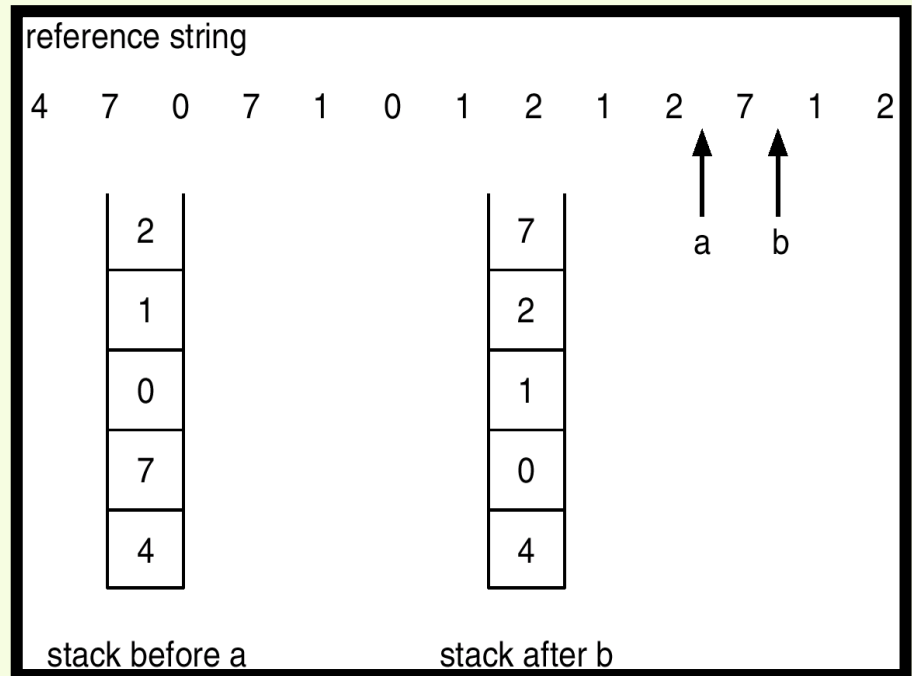
7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2		2		4	4	4	0			1		1		1		
	0	0	0		0		0	0	3	3			3		0		0		
		1	1		3		3	2	2	2			2		2		7		

Określenie porządku ramek ~LRU

Liczniki

- do każdej pozycji tablicy stron - rejestr czasu użycia
- wymiana strony z najmniejszą wartością rejestru
- wymaga przeglądania tablicy stron

Stos (dwukierunkowa lista ze
wskaźnikami do czoła i końca)
przy każdym odwołaniu do strony
jej numer wyjmuję się ze stosu
i umieszcza się na jego szczycie
(każde uaktualnienie listy
- czasochłonne)
dno stosu - wskaźnik końcowy
listy określa najdawniej
używaną stronę




LRU wymaga odpowiedniego sprzętu

Algorytmy przybliżające LRU

algorytm dodatkowych bitów odniesienia

- bit odniesienia - ustawiany przy dostępie do strony
- 8-bitowe rejestry przesuwane - odnotowanie stanu bitu odniesienia w regularnych odstępach czasu (np. co 100ms)

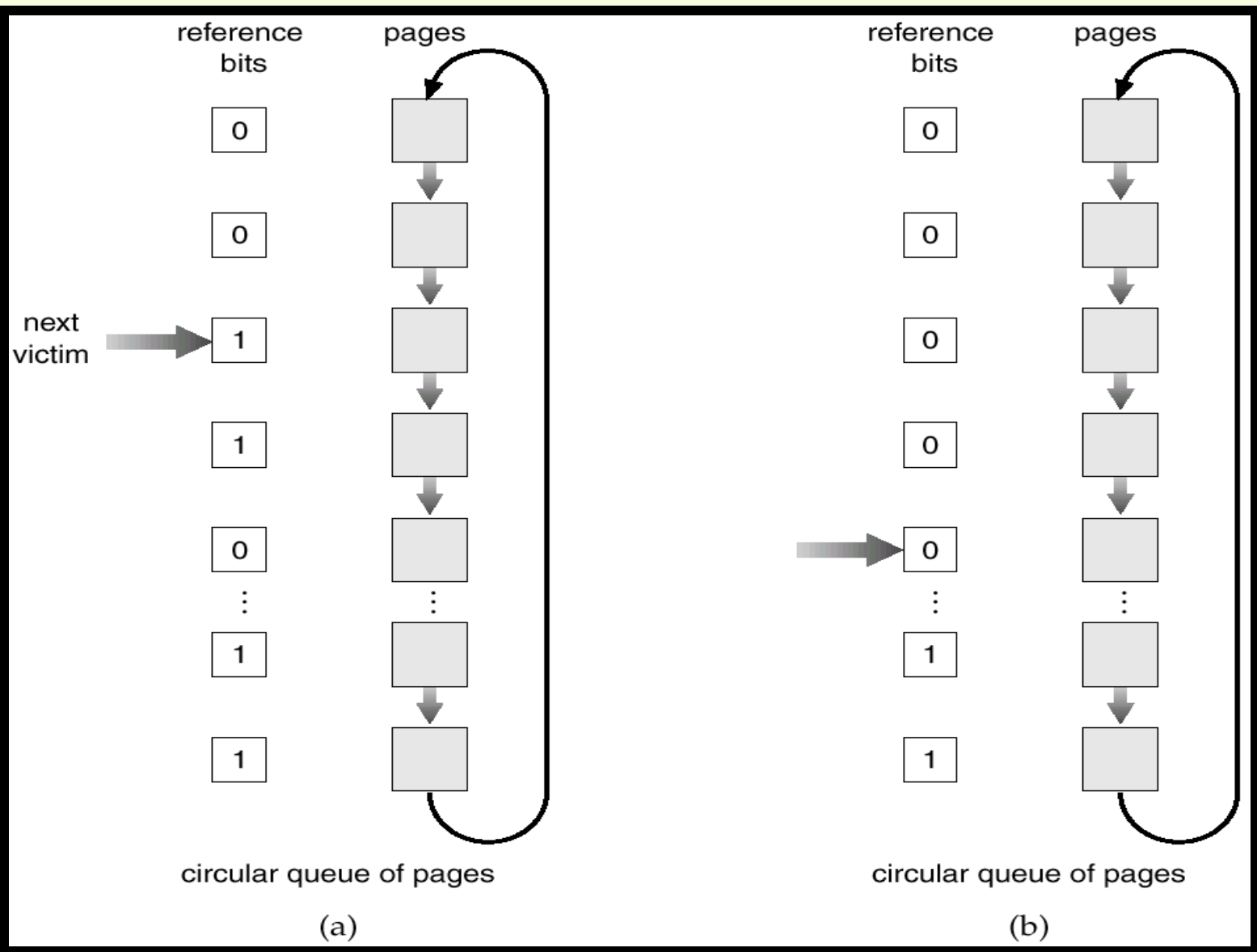

11111111 00000000 00000001 10000000

- wymieniana strona najdawniej używana (najmniejsza liczba w rejestrze)
- jeśli kilka stron ma taką samą wartość rejestru – wymiana wszystkich lub FIFO

Algorytmy przybliżające LRU

algorytm drugiej szansy (zegarowy)

- liczba bitów historii=0
- podstawą - algorytm FIFO
- po wybraniu strony - sprawdzenie bitu odniesienia;
 - jeśli 0 - strona zastąpiona;
 - jeśli 1 - „druga szansa”
- wybór kolejnej strony wg. FIFO
- „druga szansa” - zerowanie bitu odniesienia;
 - ustawienie czasu bieżącego (koniec kolejki)
- strona często eksploatowana - nigdy nie będzie zastąpiona



Algorytmy przybliżające LRU

ulepszony algorytm drugiej szansy

- wykorzystanie bitu odniesienia i bitu modyfikacji
- 4 klasy stron:
 - 1. (0,0) nie używana ostatnio, nie zmieniona (najlepsza do zastąpienia)
 - 2. (0,1) nie używana ostatnio, zmieniona (wymaga zapisu na dysk przed wymianą)
 - 3. (1,0) używana ostatnio, nie zmieniona
 - 4. (1,1) używana ostatnio, zmieniona
- zastąpienie strony z najniższej niepustej klasy

Algorytmy zliczające

wykorzystanie licznika odwołań do każdej ze stron

- algorytm LFU (*least frequently used*) –

- zastępowanie strony o najmniejszym liczniku odwołań
 - przesuwanie liczników o 1 bit w prawo co pewien czas

- algorytm MFU (*most frequently used*) –

- zał. - strona z najmniejszą wart. licznika została niedawno sprowadzona i będzie używana

kosztowna implementacja; nie przybliżają OPT

Algorytmy buforowania stron

Procedury wspomagające:

- przechowywanie puli wolnych ramek
zanim strona-ofiara zostanie usunięta
potrzebna strona czytana do wolnej ramki z puli
- przechowywanie listy zmienionych stron;
- zapis zmienionych stron na dysk przez urządzenie stronicujące w wolnym czasie
- pula wolnych ramek + inf. o tym jaka strona rezydowała w każdej ramce
(możliwość ponownego jej wykorzystania)

Przydział ramek

- minimalna liczba ramek określona przez zbiór rozkazów w architekturze komputera
(liczba adresów w rozkazie, adresowanie pośrednie..)
- maksymalna liczba ramek wynika z ilości dostępnej pamięci fizycznej

ALGORYTMY PRZYDZIAŁU m RAMEK n PROCESOM

- przydział równy - każdemu procesowi m/n ramek
- przydział proporcjonalny - każdemu procesowi a_i ramek
 s_i - wielkość pamięci wirtualnej procesu p_i

$$S = \sum s_i$$

$$a_i = s_i / S \cdot m$$

Przydział ramek

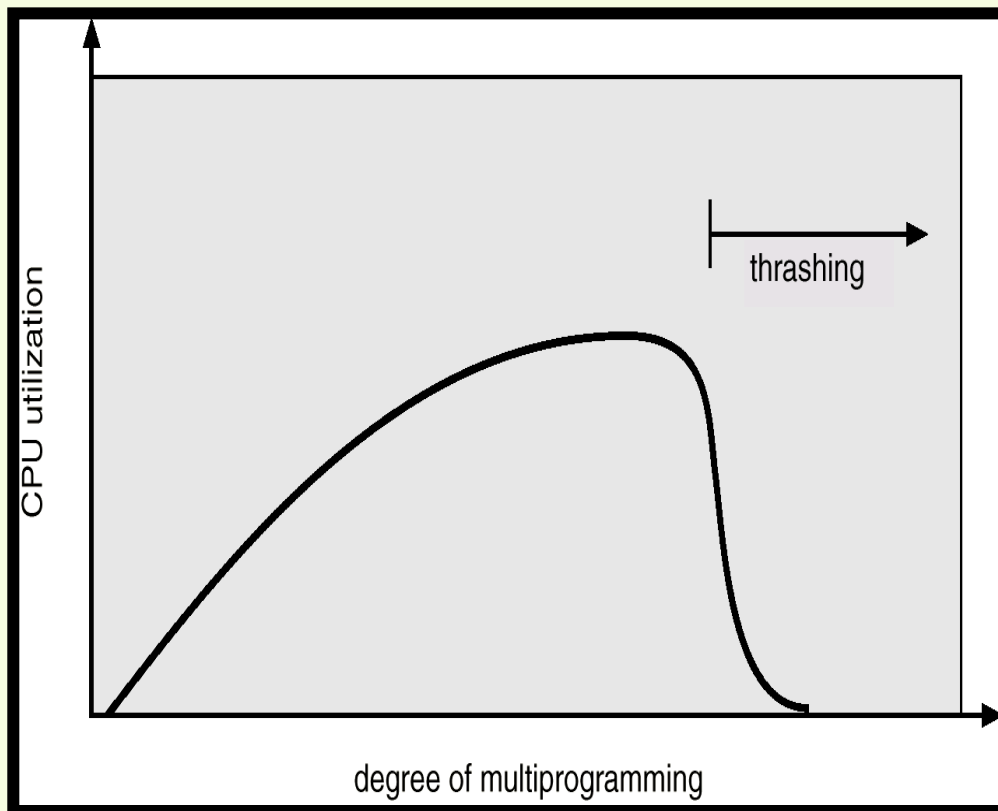
- wielkość przydziału dla każdego procesu zależy od stopnia wieloprogramowości
- procesy traktowane niezależnie od priorytetów
- przydział proporcjonalny zależnie od rozmiaru i priorytetu procesu

Przydział globalny i lokalny

- zastępowanie globalne - dla danego procesu możliwy wybór ramki z całej puli ramek (innych procesów też)
- zastępowanie lokalne - ogranicza wybór do zbioru ramek przydzielonych do danego procesu

zastępowanie globalne zapewnia lepszą przepustowość systemu
lecz zachowanie procesu silnie zależy od warunków zewnętrznych

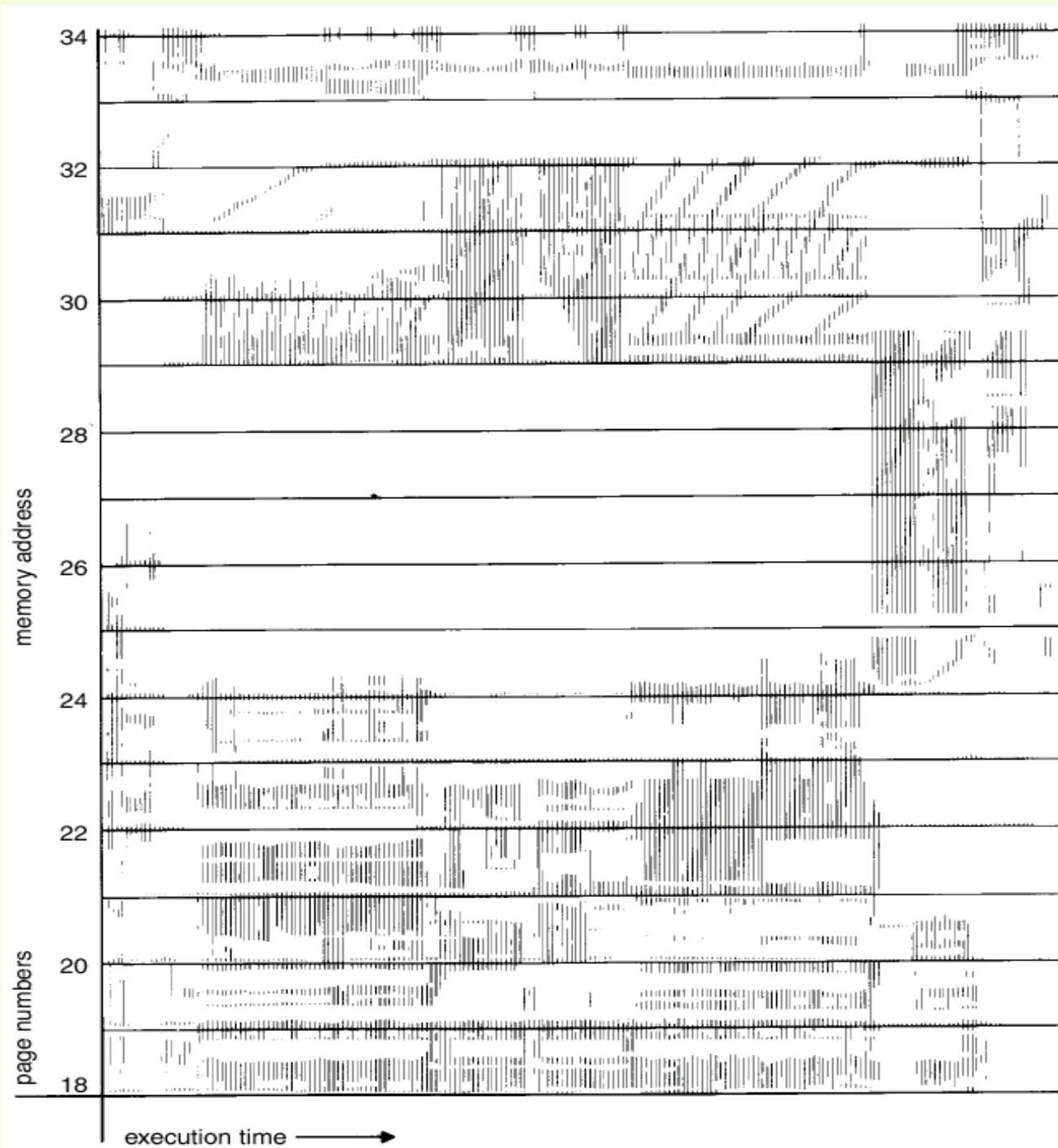
- liczba ramek niskopriorytetowego procesu $<$ minimum \Rightarrow
zawieszenie wykonania procesu
(zwolnienie wszystkich ramek - wymiana; średni poziom planowania)
- gdy proces dysponuje małą liczbą aktywnie używanych stron \Rightarrow
częste zastępowanie - **szamotanie - *thrashing***



czas stronicowania $>$ czas
wykonania

SZAMOTANIE

- zmniejszenie wykorzystania procesora ->
zwiększenie stopnia wieloprogramowości
przy globalnym algorytmie zastępowania stron:
- częste braki stron
- kolejka do urządzenia stronicującego
- opróżnienie kolejki procesów gotowych do wykonania
- zmniejszenie wykorzystania procesora
- zwiększenie stopnia wieloprogramowości
- ograniczenie efektów szamotania:
 - lokalny lub priorytetowy algorytm zastępowania
 - dostarczenie procesowi właściwej liczby ramek –
strategia tworzenia zbioru roboczego **model strefowy wykonania procesu**



model strefowy wykonania procesu

- strefa - zbiór stron pozostających we wspólnym użyciu
- strefy programu - określone przez jego strukturę i struktury danych
- gdy przydzielono mniej ramek niż wynosi rozmiar strefy - szamotanie

model zbioru roboczego - zał. program ma charakterystykę

strefową; okno zbioru roboczego - Δ ostatnich odniesień do stron

zbiór roboczy - zbiór stron, do których nastąpiło Δ ostatnich odniesień-
przybliża strefę programu

$Z = \sum RZ R_i$ (Z - zapotrzebowanie na ramki;

$RZ R_i$ - rozmiar zbioru roboczego procesu i)

$Z > m$ - szamotanie

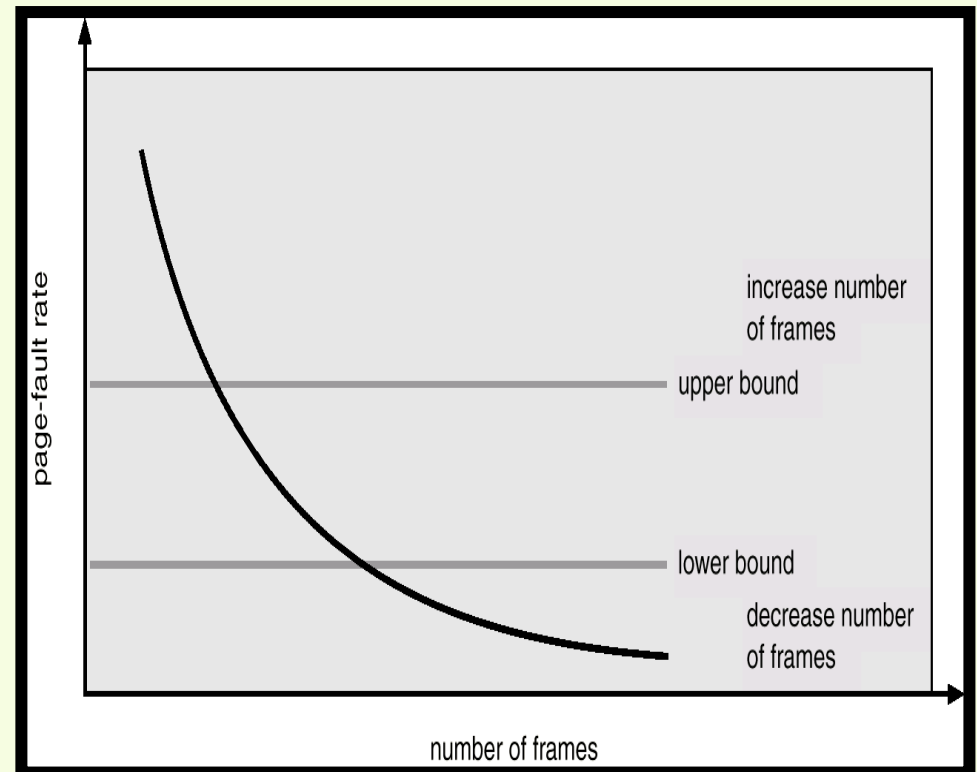
Model zbioru roboczego

optymalizacja wykorzystania procesora:
wysoki stopień wieloprogramowości
zapobieganie szamotaniu

- SO przydziela każdemu procesowi tyle ramek, ile wymaga rozmiar jego zbioru roboczego
- jeśli są wolne ramki - rozpoczęcie nowego procesu
- gdy suma rozmiarów zbiorów roboczych > dostępne ramki -> wstrzymanie wybranego procesu
(usunięcie z pamięci, przydzielenie ramek innym procesom, późniejsze wznowienie)
- implementacja - bit odniesienia + przerwania zegara

Częstość braków stron

- mierzenie częstości braków stron procesów PFF (*page-fault-frequency*)
- zapobieganie szamotaniu
- ustalenie górnej i dolnej granicy pożądanego poziomu braków stron
- przydzielenie dodatkowej ramki lub usunięcie ramki, gdy liczba braków stron przekroczy limity
- dodatkowo możliwość wstrzymania procesu



STRONICOWANIE WSTĘPNE

- wstępna faza procesu (po wznowieniu)
 - wysoka aktywność stronicowania
- stronicowanie wstępne (*prepaging*)
- dla każdego procesu - lista stron zbioru roboczego;
przy wznowieniu przeniesienie zbioru roboczego do pamięci

ROZMIAR STRONY

512 (2^9) - 16384 (2^{14})

- duże strony:

- wielkość tablicy stron każdego procesu

pam. wirt. 4MB - 4096 stron 1024B lub 512 stron 8192B

- czas operacji we/wy

czas wyszukiwania ($20+8\text{ms}$) > czas przesyłania ($2\text{MB/s} \rightarrow 512\text{B} - 0.2\text{ms}$)

1024B – 1 strona – 28.4ms

2 strony – 56.4ms

- liczba braków stron

- małe strony

- lepsze wykorzystanie pamięci (fragmentacja wewnętrzna)

- mniejsza ilość przesyłanych informacji - lepsza rozdzielczość,
dopasowanie do stref programu

Problemy odwróconej tablicy stron

- Oszczędność miejsca w pamięci fizycznej używanej do translacji
- Wymagana dodatkowa tablica stron dla każdego procesu –
odwołania do niej jedynie przy braku strony
- Zewnętrzne tablice stron –
w pamięci wirtualnej – podlegają stronicowaniu

Struktura programu

- przetwarzanie tablicy wierszami lub kolumnami
zależnie od jej rozmieszczenia w PAO
- wybór języka programowania a lokalność
odwołań do pamięci
- kompilacja, ładowanie – odseparowanie
kodu i danych; rozmieszczenie procedur na stronach

Blokowanie stron

- dla każdej ramki – bit blokowania
(wyłączenie ze stronicowania)
 - przed op. zapisu bloku na taśmę –
blokowanie wszystkich zawierających go stron
 - blokowanie właśnie sprowadzonych, jeszcze
nieużywanych stron niskopriorytetowego procesu
- brak wyzerowania bitu blokowania –
zablokowanie ramki

SEGMENTACJA NA ŻĄDANIE

- Stronicowanie na żądanie wymaga sporych ilości sprzętu
- OS/2 Intel 80286 operuje segmentami
- deskryptor segmentu: długość, tryb ochrony, położenie, bit poprawności, bit udostępnienia
- kolejka deskryptorów wszystkich segmentów w pamięci;
 - na początku segmenty z ustawionym bitem udostępnienia (zerowane)
- pułapka - spr. czy dostępna wolna pamięć pomieści segment
 - (tak) upakowanie
 - (nie) wymiana segmentu z końca kolejki
 - »jeśli wystarczy miejsca
 - uaktualnienie deskryptora, na początek kolejki
 - »jeśli nadal mało miejsca - upakowanie.....

Strategie przydziału pamięci

Wymiany

–Systemy ze stronicowaniem

- Najdawniej używana
- Najmniej używana
- Najdawniej załadowana

–Systemy bez stronicowania

Pobierania

- na żądanie
- z wyprzedzeniem

Rozmieszczenia

–Systemy bez stronicowania

- Najlepsze dopasowanie $x_1 < x_2 < \dots < x_n$
- Najgorsze dopasowanie $x_1 > x_2 > \dots > x_n$
- Pierwsze dopasowanie -
uporządkowanie względem adresów bazowych
przeszukiwanie listy od ostatniego wskaźnika
- Algorytm bliźniaków

–Systemy ze stronicowaniem

Algorytm bliźniaków

rozmiar segmentu $s = 2^i$; $i < k$

lista(i)

procedure szukaj(i)

begin if $i \geq k+1$ **then** błąd;

if lista(i) pusta **then**

begin

szukaj(i+1);

podziel;

umieść w lista(i)

end;

pobierz pierwszy el. lista(i)

end;

UNIX

- przed 3BSD – wymiana
(obsługa średnioterminowego planowania)
 - przydział ciągłego obszaru pamięci
 - pierwsze dopasowanie (*first fit*)
 - wzrost rozmiaru pamięci procesu –
przekopiowanie w nowe miejsce;
szukanie obszaru na końcu zajmowanego
 - przy braku wystarczającego obszaru – proces odsyłany na dysk
 - zapobieganie szamotaniu – minimalny czas pozostawania w pamięci
 - dzielone segmenty tekstu nie są odsyłane na dysk ani
ponownie wczytywane

UNIX

- stronicowanie na żądanie
 - możliwość odzyskania strony z listy wolnych ramek
 - wstępne stronicowanie + lista wolnych ramek
 - blokowanie stron podczas przesyłania
 - algorytm zastępowania stron –
zmodyfikowany algorytm drugiej szansy
- planowanie przydziału procesora,
wymiana obszarów pamięci i stronicowanie
 - im niższy priorytet procesu - bardziej prawdopodobne
usunięcie strony i wyrzucenie z pamięci w całości
- strony sprzętowe (VAX) = 512 B
 - są grupowane po dwie, aby operacje we/wy
były wydajne

Linux

- zarządca podstawowej pamięci fizycznej
 - dyspozytor stron
 - algorytm sąsiednich stert
(może przydzielić na zamówienie partie stron fizycznie sąsiadujące)
 - usługa **kmalloc** - przydziela na żądanie całe strony,
następnie dzieli je na mniejsze kawałki
- obsługa pamięci wirtualnej
 - brak wymiany; jedynie mechanizm stronicowania
 - algorytm postępowania (które strony zapisywać na dysk i kiedy)
zmodyfikowana wersja algorytmu drugiej szansy (LFU)
 - mechanizm stronicowania (realizuje przesłania)

Windows NT

- Zarządca pamięci wirtualnej VM - część egzekutora NT
- odwzorowanie pam. wirt - pam. rzecz. realizowane sprzętowo
- stronicowanie; strona 4 kB
- plik stronicowania na dysku
- adresy 32-bitowe (każdy proces - przestrzeń 4GB)
- górne 2 GB - używane w trybie jądra;
 identyczne dla wszystkich procesów
- algorytm zastępowania stron –
 FIFO w odniesieniu do każdego procesu
- prowadzenie stron przyległych - lokalność odwołań do pamięci
- początkowo proces otrzymuje 30 stron w zbiorze
 roboczym (zmniejszanie o 1)

ZARZĄDZANIE OBSZAREM WYMIANY

- cel – najlepsza przepustowość pamięci wirtualnej
- systemy z wymianą - obraz całych procesów
- systemy ze stronicowaniem – strony
- wiele obszarów wymiany – różne dyski
- nadmierne oszacowanie wielkości obszaru wymiany - bezpieczniejsze

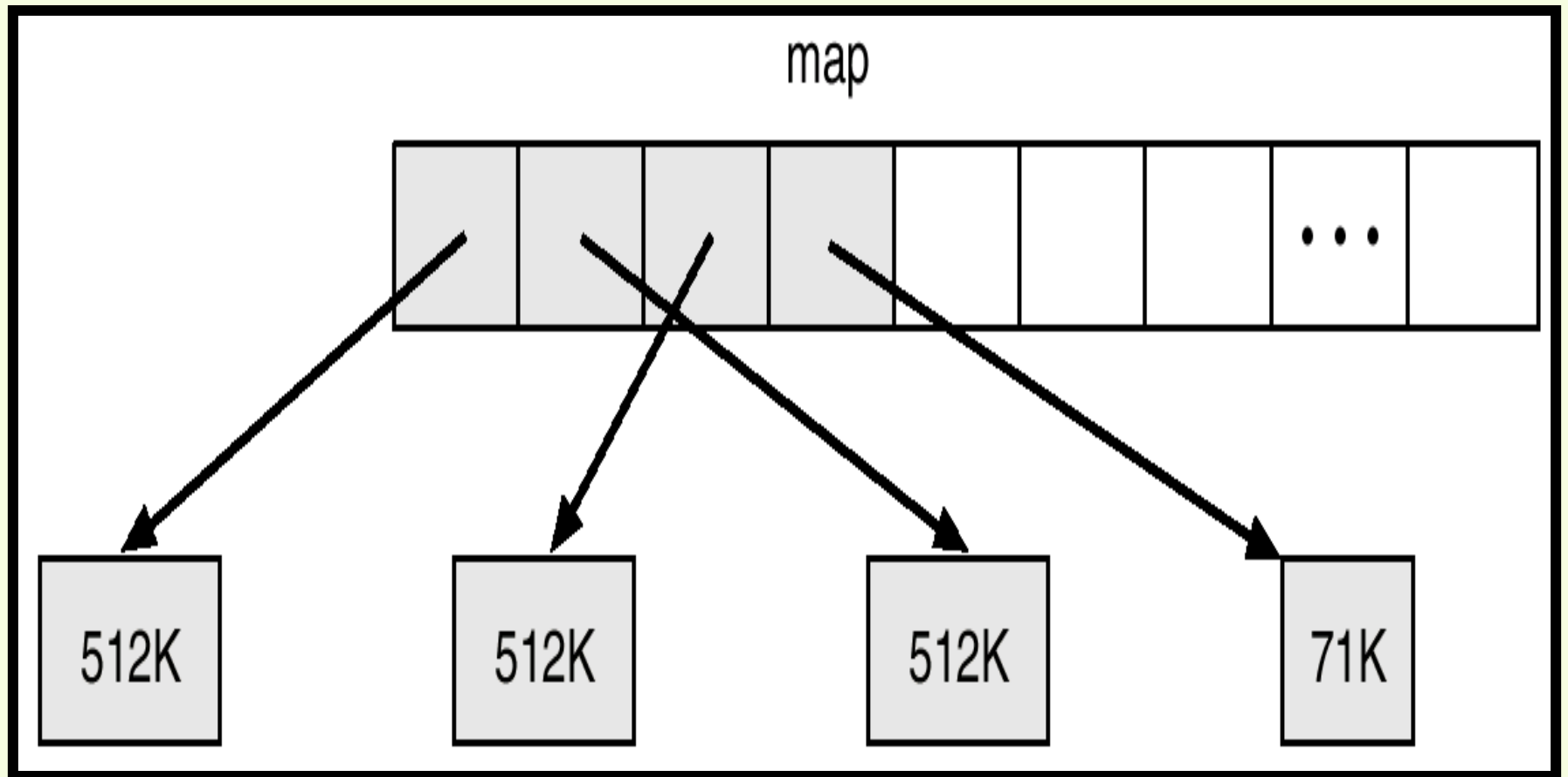
Umiejscowienie obszaru wymiany

- system plików
 - zastosowanie procedur systemu plików
 - mała wydajność
- osobna strefa dyskowa
 - bez struktury katalogowej
 - Zarządca pamięci obszaru wymiany –
optymalizacja szybkości

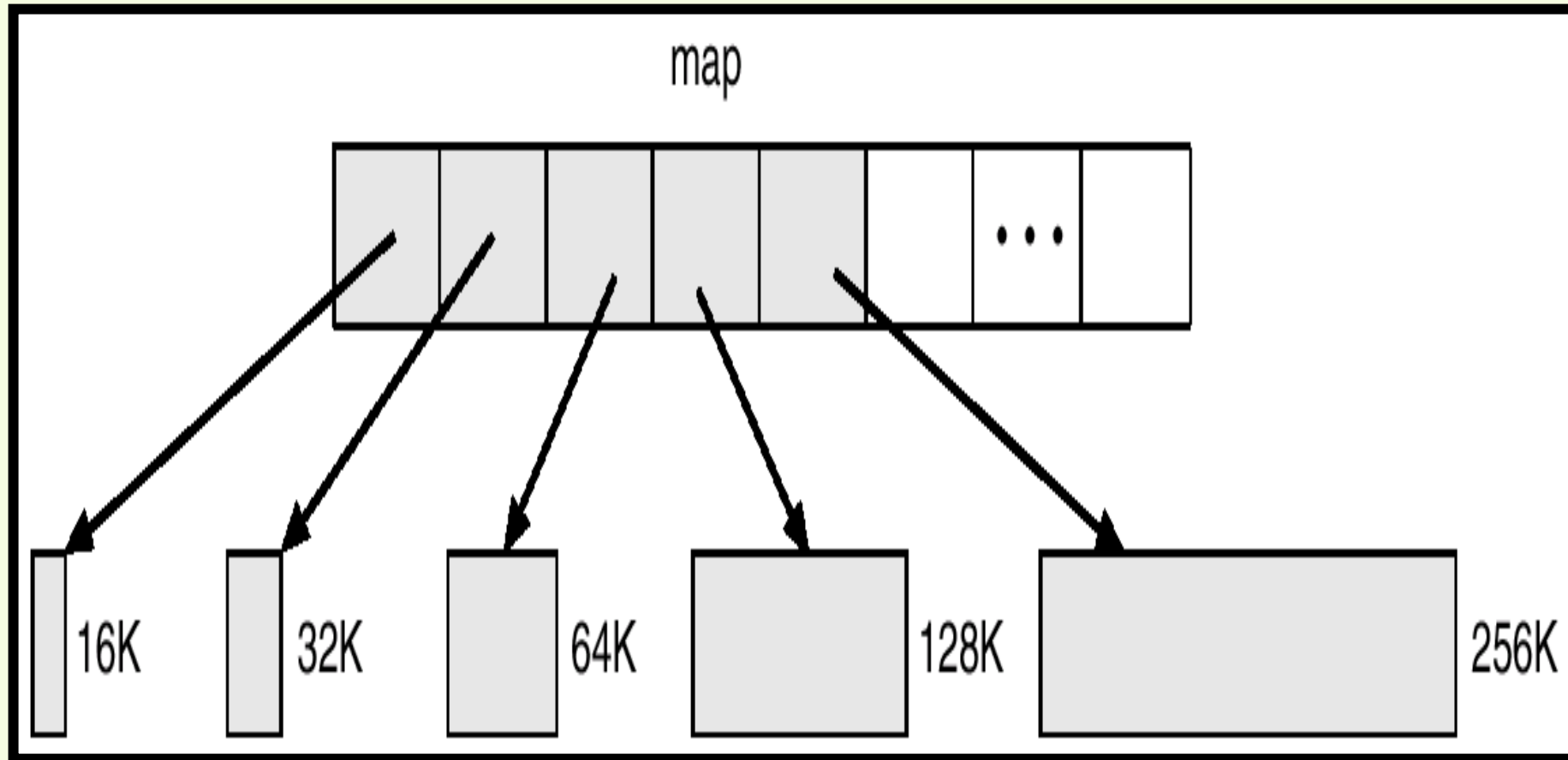
obszar wymiany systemu UNIX

- 4.3 BSD – uruchomiony proces –
przysłał obszar wymiany:
 - strony tekstu (sprowadzone z systemu plików przy starcie procesu) po 512kB + końcówka nkB
 - segment danych – czytane z systemu plików lub tworzone bloki o zmiennych rozmiarach
- Wspólne użytkowanie identycznych stron w kilku procesach zarówno w pamięci fizycznej jak i obszarze wymiany
- Jądro używa 2 procesowe **mapy wymiany**

mapa wymiany segmentu tekstu



mapa wymiany segmentu danych



rozmiar bloku $i = 2^i \cdot 16\text{KB}$; rozmiar $\leq 2\text{MB}$

proces powiększa segment danych – przydziela mu się nowy blok, 2 x większy

Intel

- Tablice deskryptorów: LDT (Local Descriptor Table), GDT (Global...)
- LDT – lokalne dla procesu segmenty kodu, danych, stosu
- GDT – segmenty systemowe (m. in. tablice LDT)
- Segmenty stronicowane
- Strona - 4 kB
- Stronicowanie – 2poziomowe
- Przestrzeń adresowa 1 segmentu – 4GB (32 bity) -
 2^{20} stron 4kB \rightarrow 2^{20} wpisów 4B
 Tablica stron - 4MB \rightarrow stronicowanie 2-poziomowe
- Adres logiczny:
 - selektor segmentu (16 bitów) - ładowany do rej. segmentowego
 CS (kod), DS (dane), SS (stos)
 - 13 b - indeks segmentu w tablicy deskryptorów
 - 1 b 1/0 – LDT/GDT
 - 2 b – poziom uprzywilejowania (0-3)
 - offset (32 bity)

- Selektor segmentu – wskazuje na deskryptor w tablicy GDT lub LDT

- Deskryptor – 8 bajtów – w tym m. in.

 - adres bazowy (32 bity)

 - limit (20 bitów)

- Porównanie przesunięcia (z adresu – offset) z limitem (z deskryptora)

 - if (przesunięcie > limit) then wyjątek

 - granulacja – limit wyrażony w bajtach lub stronach=4kB

- Adres liniowy = adres bazowy + przesunięcie

- Stronicowanie:

- Adres liniowy:

- 10 bitów – 1 z 1024 pozycja w katalogu tablic

- 10 bitów – indeks pozycji w tablicy stron

- Początek strony w PAO

- 12 bitów – pozycja na stronie

- Cache – przechowuje wartości adresów fizycznych dla ostatnio używanych:

- katalog – strona (*TLB Table Lookahead Buffer* – 32 pozycje)

- Bez stronicowania (wyzerowanie bitu rejestru sterującego procesora CR0) – adres liniowy – adresem fizycznym

- segmenty mogą nakładać się na siebie

- Bez segmentacji – wszystkie deskryptory segmentów
 - pole bazy = 0; długość = max

- Płaski model zarządzania pamięcią – Win32

- IBM OS/2 – Intel, segmentacja + stronicowanie

Adres 32-bitowy:

bity 31-22 (10) - wskaźnik pozycji w katalogu tablic stron

bity 21-12 (10) - wskaźnik pozycji w tablicy stron

bity 11-0 (12) - adres na stronie

strona $2^{12}=4\text{kB}$

każdy proces ma:

- katalog tablic stron (1024 pozycje 4-bajtowe)

każda pozycja katalogu tablic stron wskazuje na:

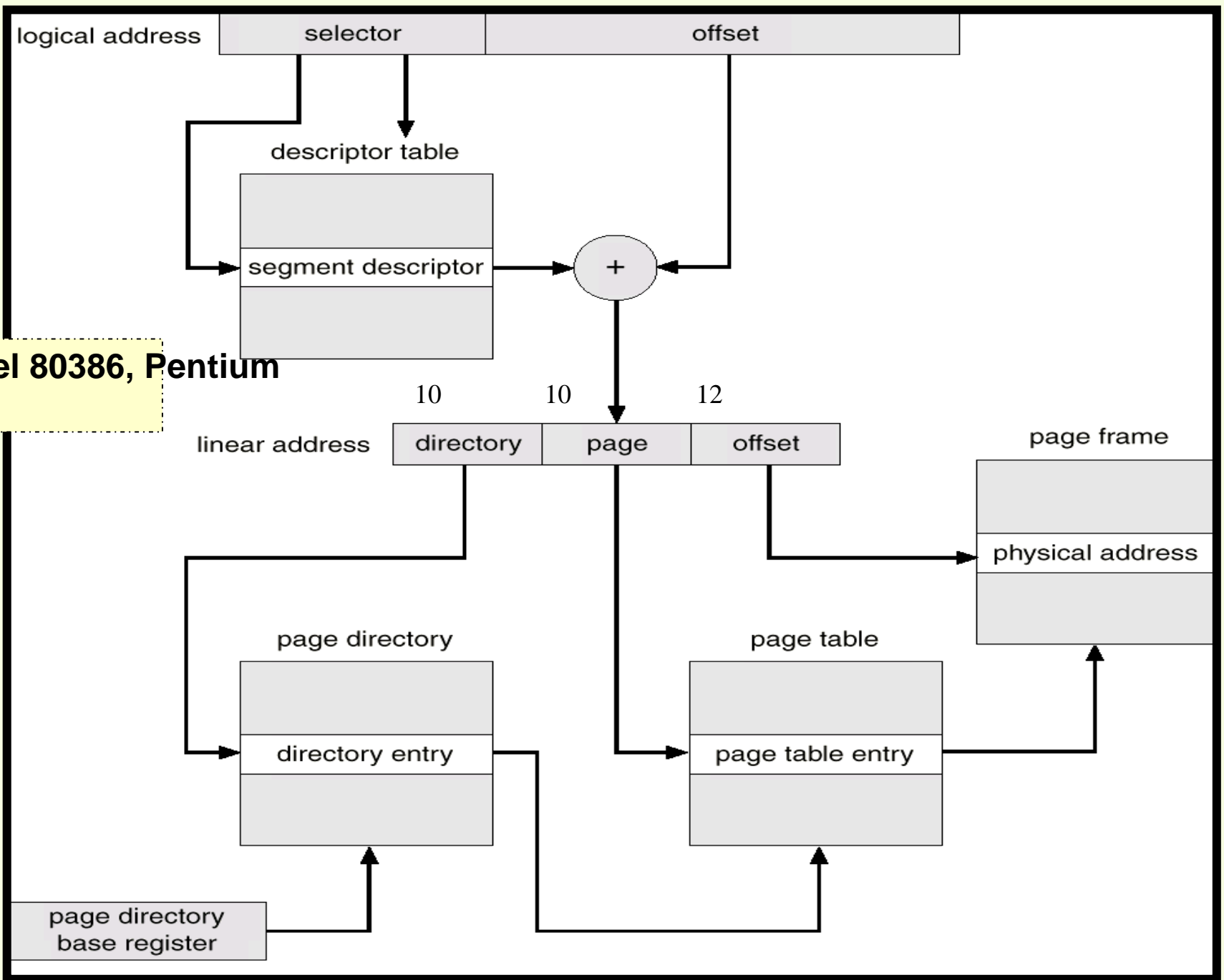
- tablicę stron (1024 pozycje 4-bajtowe)

20 bitów wykorzystanych do tworzenia adresu
(+12 najmłodszych bitów)

12 bitów nadmiarowych –

opis strony: tryb ochrony, plik stronicowania, stan strony

Intel 80386, Pentium



Ochrona dostępu do pamięci

Pentium:

Poziom segmentacji

- Deskryptor zawiera w polu DPL (*Descriptor Privilege Level*) liczbę $\langle 0;3 \rangle$
 - 0 - najwyższe uprzywilejowanie
 - 3 - najniższe (brak restrykcji dostępu)
- DPL z rejestru CS - poziom uprzywilejowania selektora wskazującego na aktualnie wykonywany kod
=poziom uprzywilejowania procesu = CPL (*Current Privilege Level*)
- $DPL \geq CPL$ - proces ma dostęp do danych
(mniej lub tak samo uprzywilejowane segmenty)
- $DPL \leq CPL$ - proces może przekazywać sterowanie do tych segmentów
(bardziej lub tak samo uprzywilejowane segmenty)

Poziom stronicowania

- Strony: użytkowe, systemowe
- Blokada modyfikacji stron
- Poziomy uprzywilejowania - 0, 1, 2, 3, jądro
- Windows - wykorzystuje poziomy 0 i 3

Pamięć wirtualna a cache'owanie

- Pamięć wirtualna – program w pamięci dyskowej;
niektóre strony – w ramkach (PAO)
 - Cache – kod w PAO, fragmenty kodu – w pamięci podręcznej
 - Obsługa braku stron – zadanie SO
 - Obsługa chybienia pamięci podręcznej – sprzętowa
 - Rozmiar: bloki cache – 64 B
strony – 4kB
 - Odwzorowanie adresów:
stronicowanie – wykorzystuje najbardziej znaczące
bity adresu wirtualnego
- Cache – wykorzystuje najmniej znaczące bity
adresu fizycznego