Systemy operacyjne III

WYKŁAD 5

Jan Kazimirski

Pamięć wirtualna

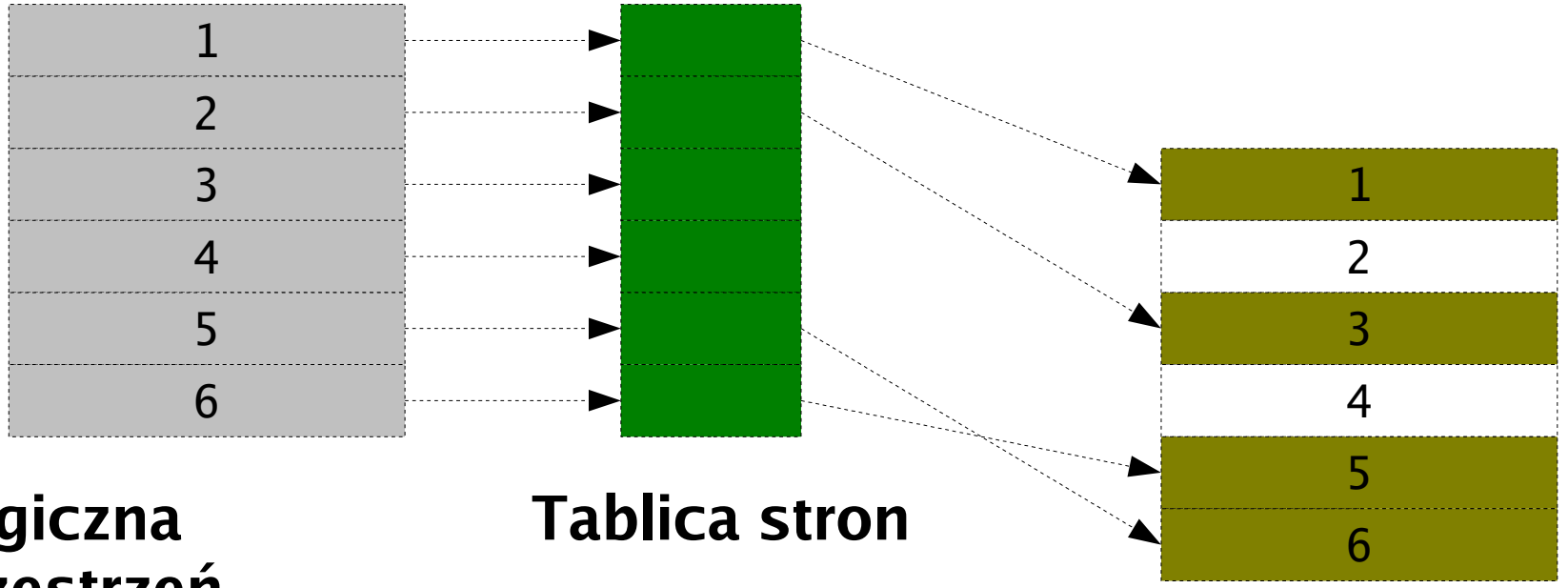
Stronicowanie

- Pamięć podzielona na niewielki bloki
- Bloki procesu to **strony** a bloki fizyczne to **ramki**
- System operacyjny przechowuje dla każdego procesu osobną tablicę stron
- Tablica stron odwzorowuje logiczne strony na fizyczne ramki

Stronicowanie c.d.

- Adresy fizyczne i logiczne nie są związane
- Ciągłe bloki logiczne mogą być mapowane na rozłączne obszary fizyczne.
- Część stron może nie być wcale mapowana na pamięć fizyczną.
- Adresy logiczne procesów mogą się dublować (inne adresy fizyczne).
- Różne strony logiczne mogą być mapowane na tą samą ramkę

Stronicowanie c.d.

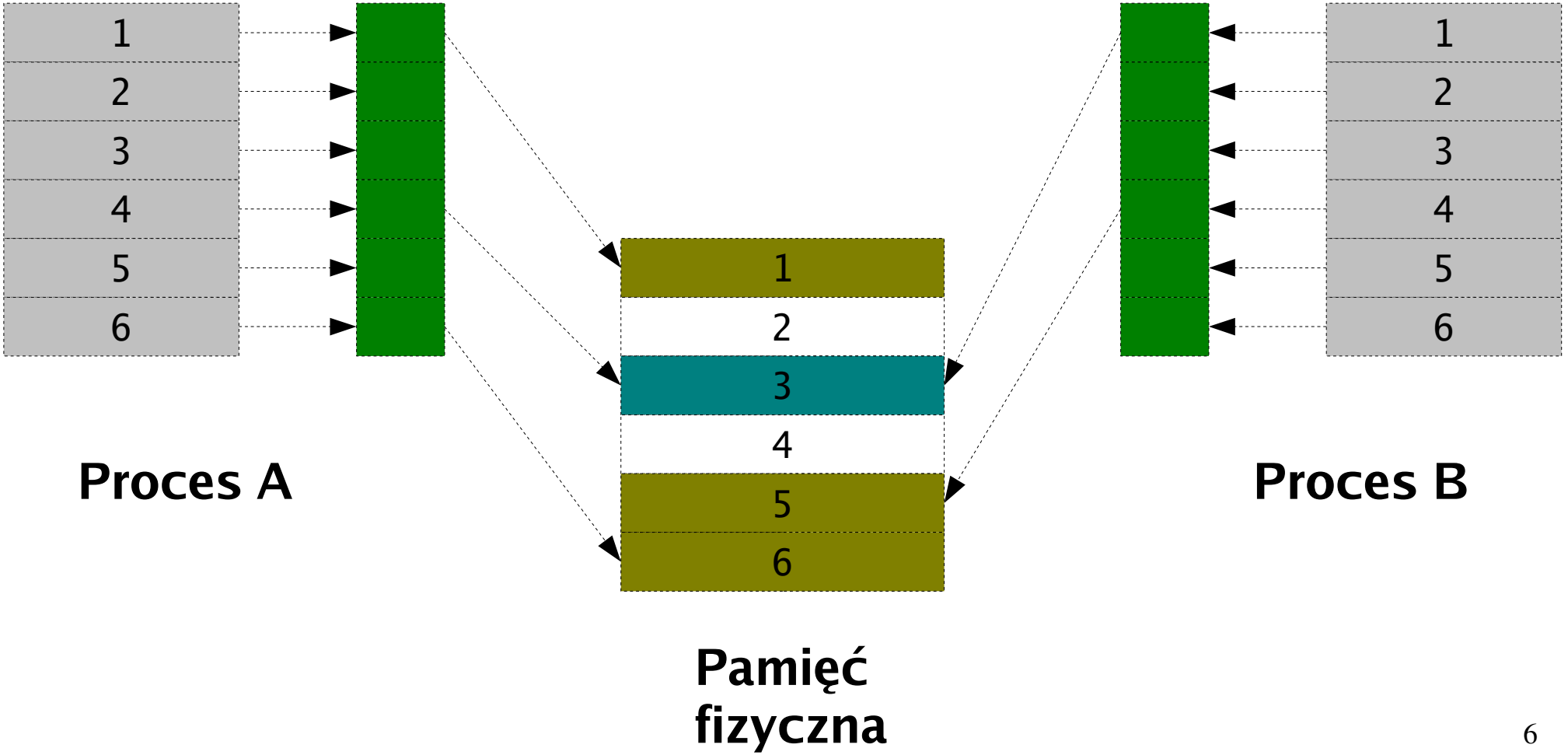


Logiczna przestrzeń adresowa procesu (strony)

Tablica stron

Przestrzeń fizyczna - ramki

Stronicowanie c.d.

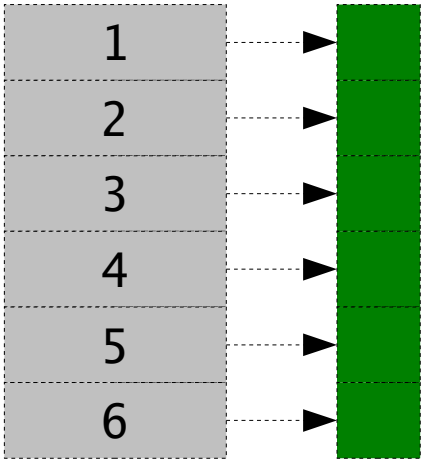


Uruchamianie Procesu

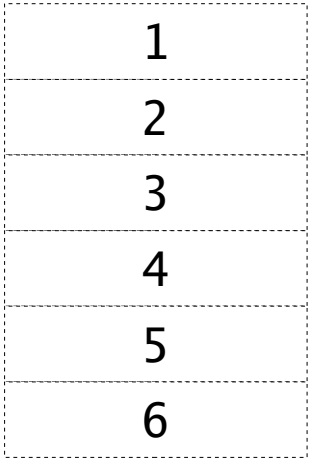
Utworzenie logicznej przestrzeni adresowej dla procesu.

Utworzenie tablicy stron

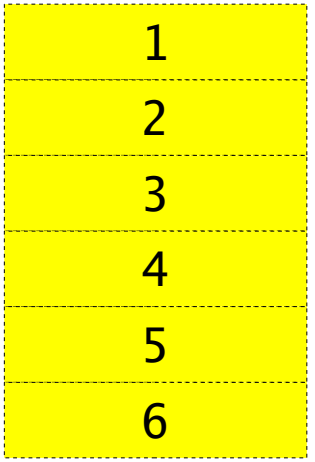
Pomimo „posiadania” swojej przestrzeni adresowej proces nie zajmuje jeszcze ani jednego bajta pamięci fizycznej poza obszarem tablicy stron



Strony logiczne



Ramki fizyczne

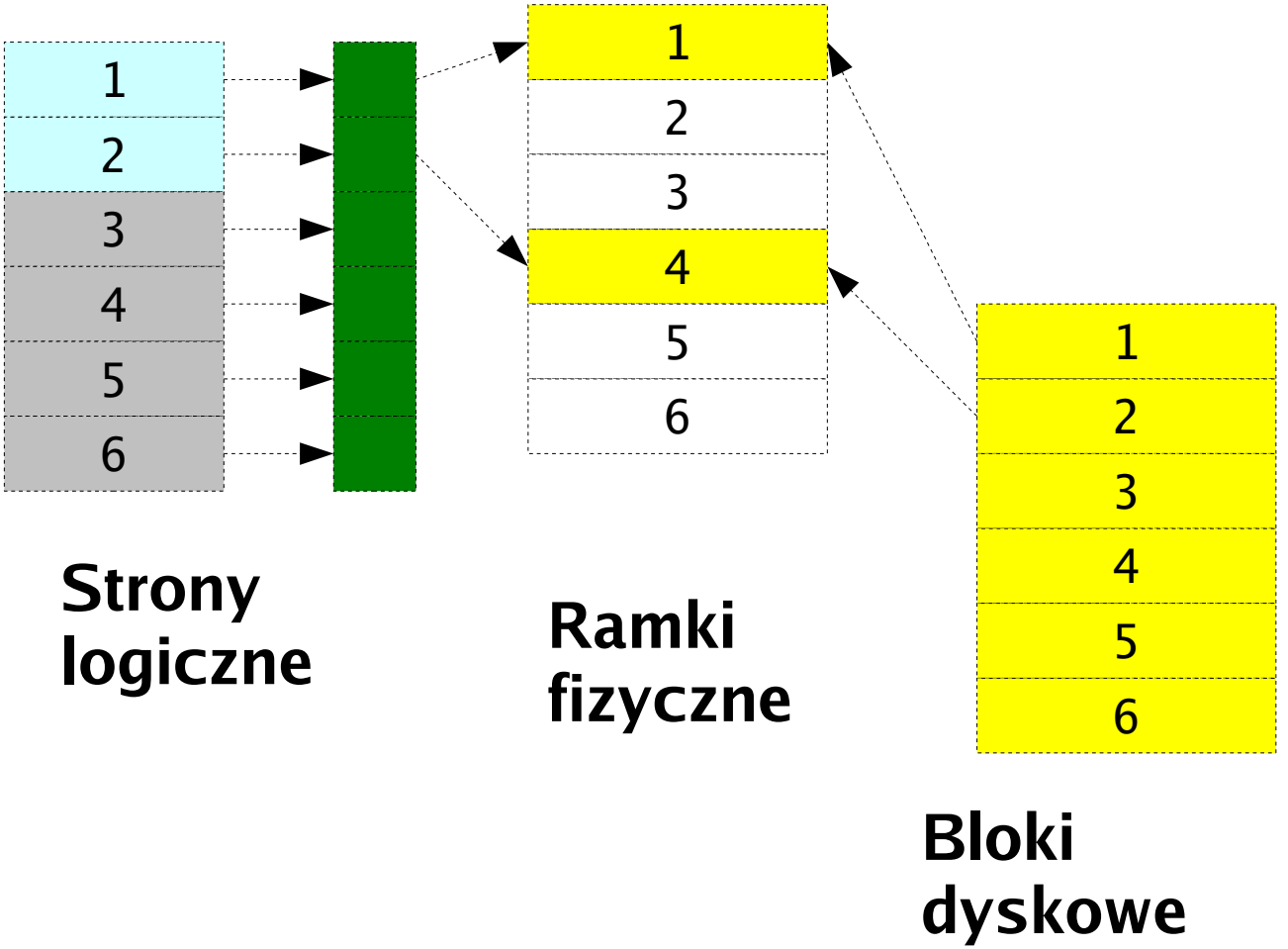


Bloki dyskowe

Uruchamianie procesu (2)

System przydziela fizyczne ramki do kilku pierwszych stron programu i ładuje fragment programu z dysku.

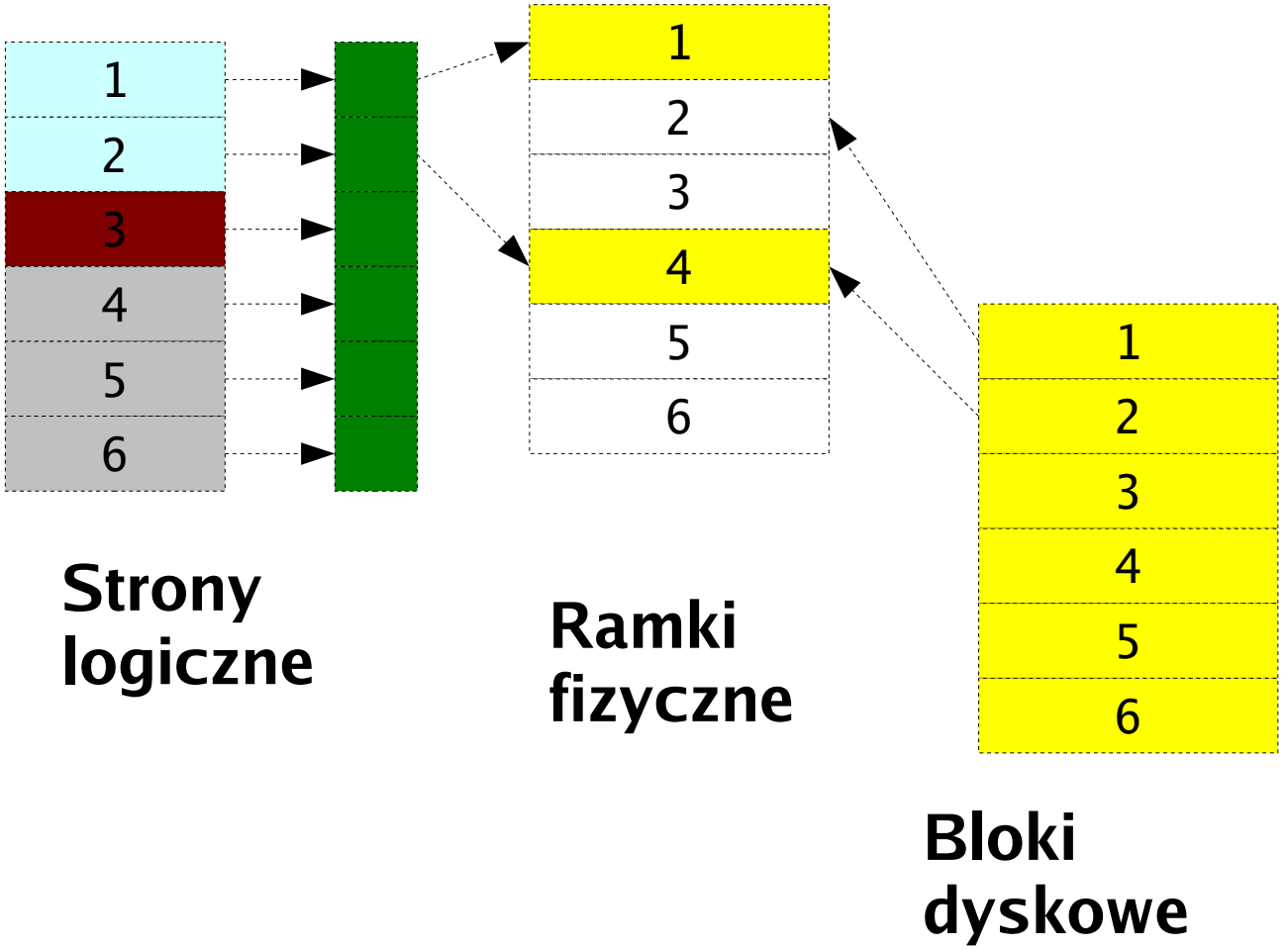
Rozpoczyna się wykonywanie programu



Wykonywanie procesu

W trakcie wykonania proces może sięgnąć do strony, która nie została załadowana do pamięci.

Następuje „Błąd braku strony” - **PAGE FAULT**

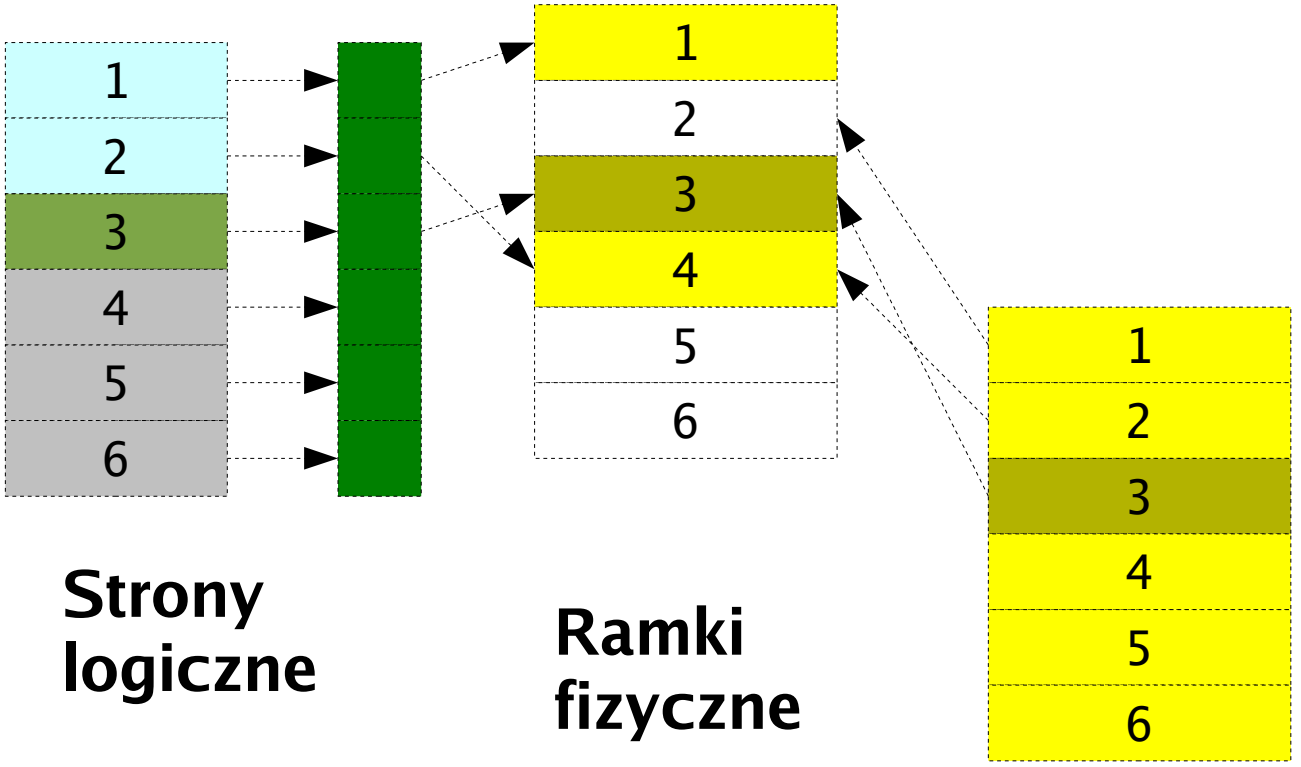


Wykonywanie procesu (2)

Po wystąpieniu błędu braku strony proces zostaje wstrzymany.

System operacyjny ładuje z dysku odpowiedni fragment i mapuje go na odpowiednią stronę

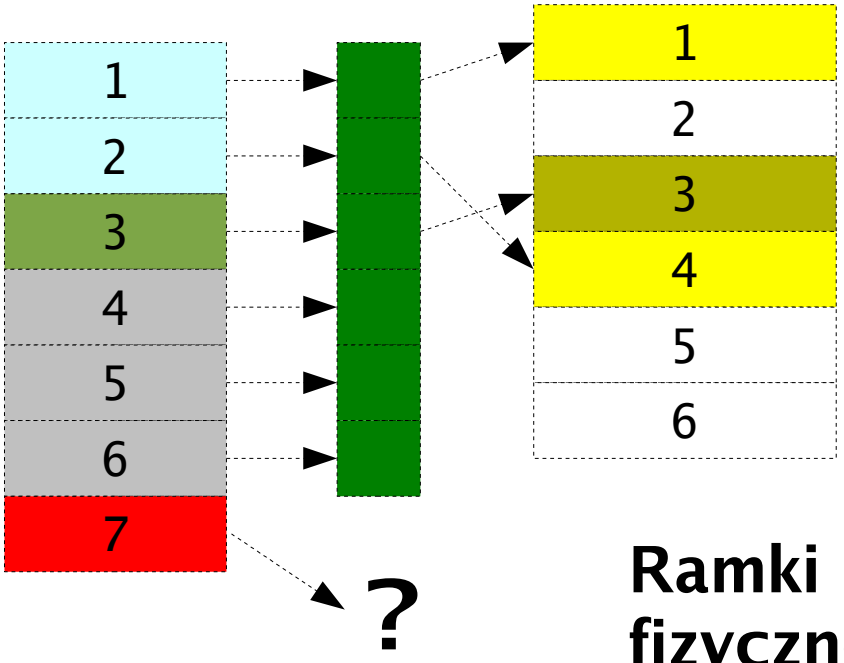
Po zakończeniu operacji dyskowej proces jest budzony i kontynuowany



Wykonywanie procesu (3)

Mechanizm stronicowania pozwala zapewnić ochronę przestrzeni adresowej procesów

Jeżeli proces próbuje uzyskać dostęp do pamięci poza swoją przestrzenią adresową, to następuje **błąd ochrony pamięci.**



Nowy proces - fork

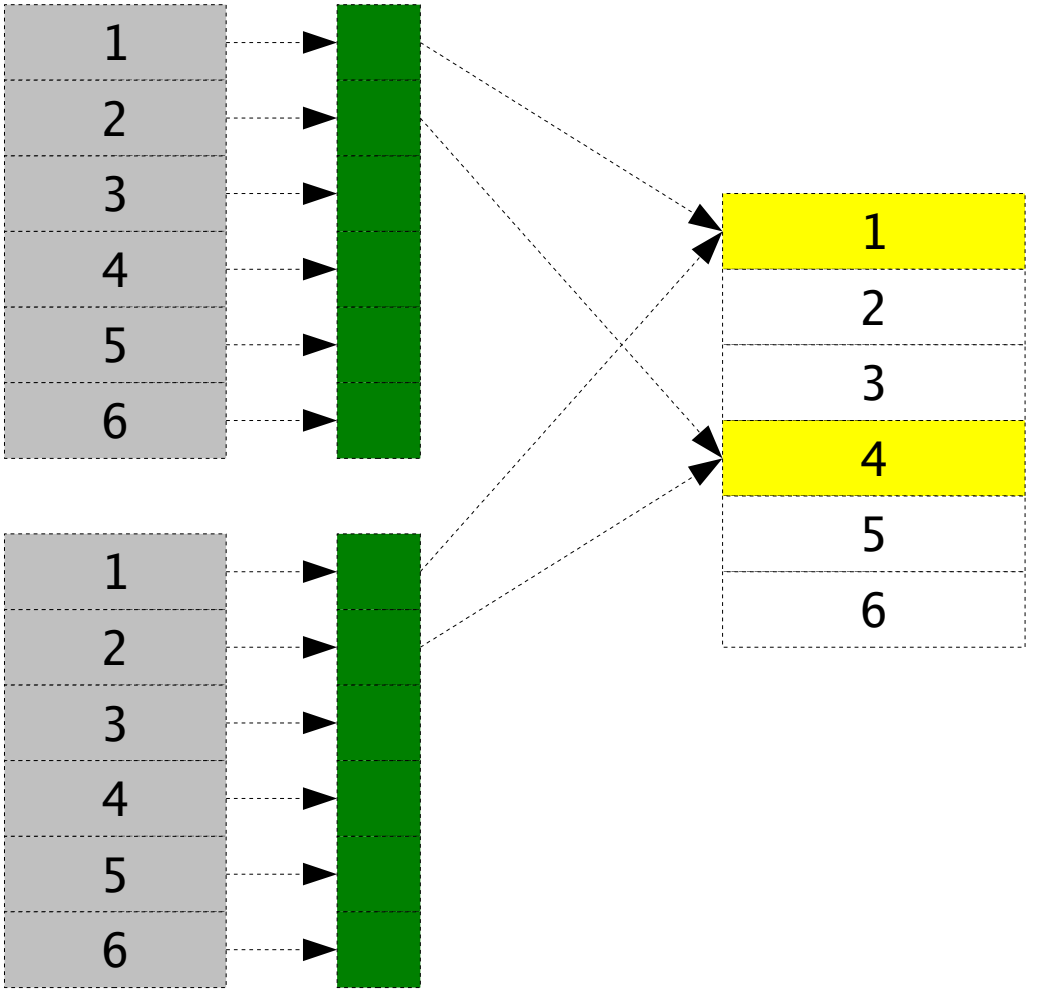
Tworzenie nowego procesu w systemach Unix/Linux to tzw. fork-and-exec.

W pierwszym etapie Wywoływana jest funkcja systemowa fork.

Tworzona jest kopia przestrzeni adresowej procesu rodzica.

Przestrzenie adresowe obu procesów mapowane są na te same ramki pamięci fizycznej

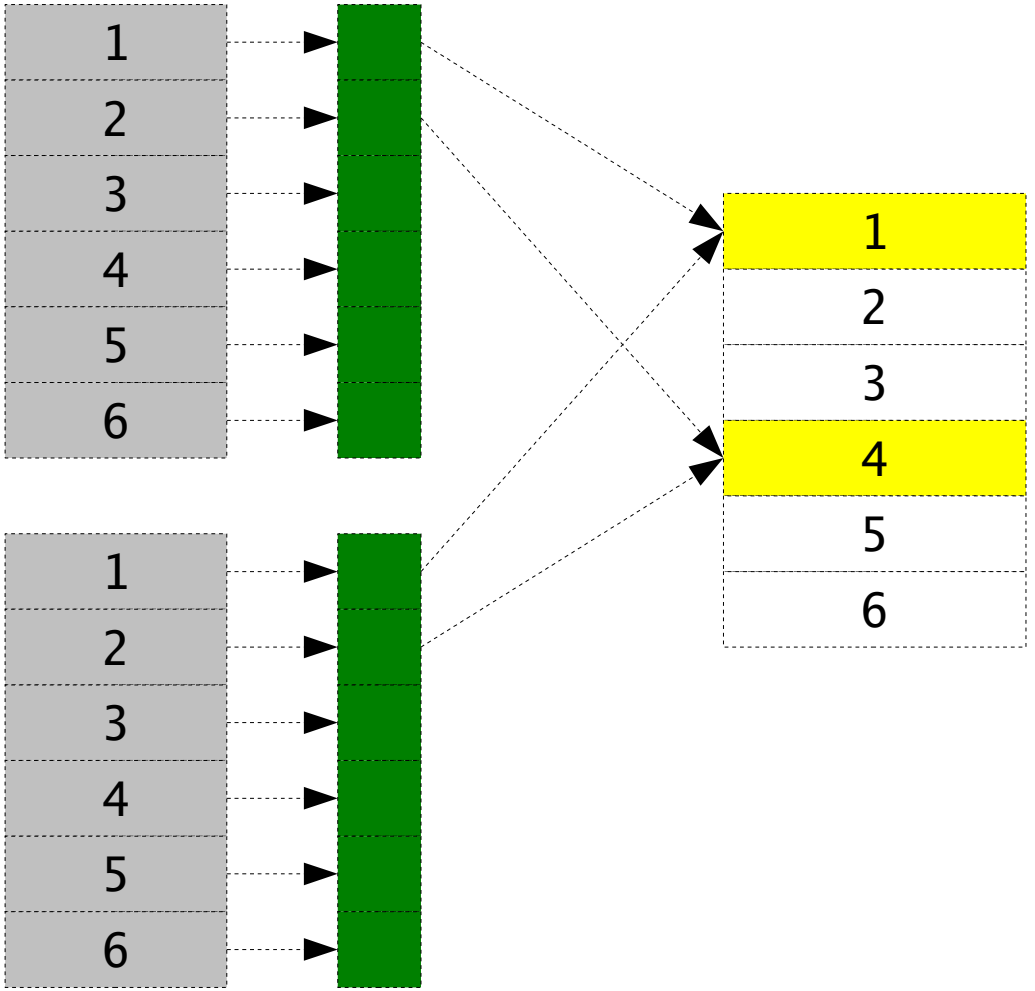
Nowy proces poza tablicą stron nie zajmuje w ogóle pamięci fizycznej



Nowy proces – fork (2)

W trakcie wykonania oba procesy mogą korzystać z tej samej ramki pamięci jeżeli realizują na niej tylko operacje odczytu

Każdy proces widzi tę ramkę jako stronę swojej własnej przestrzeni adresowej, ale fizycznie zajmują one tylko jedną ramkę w pamięci



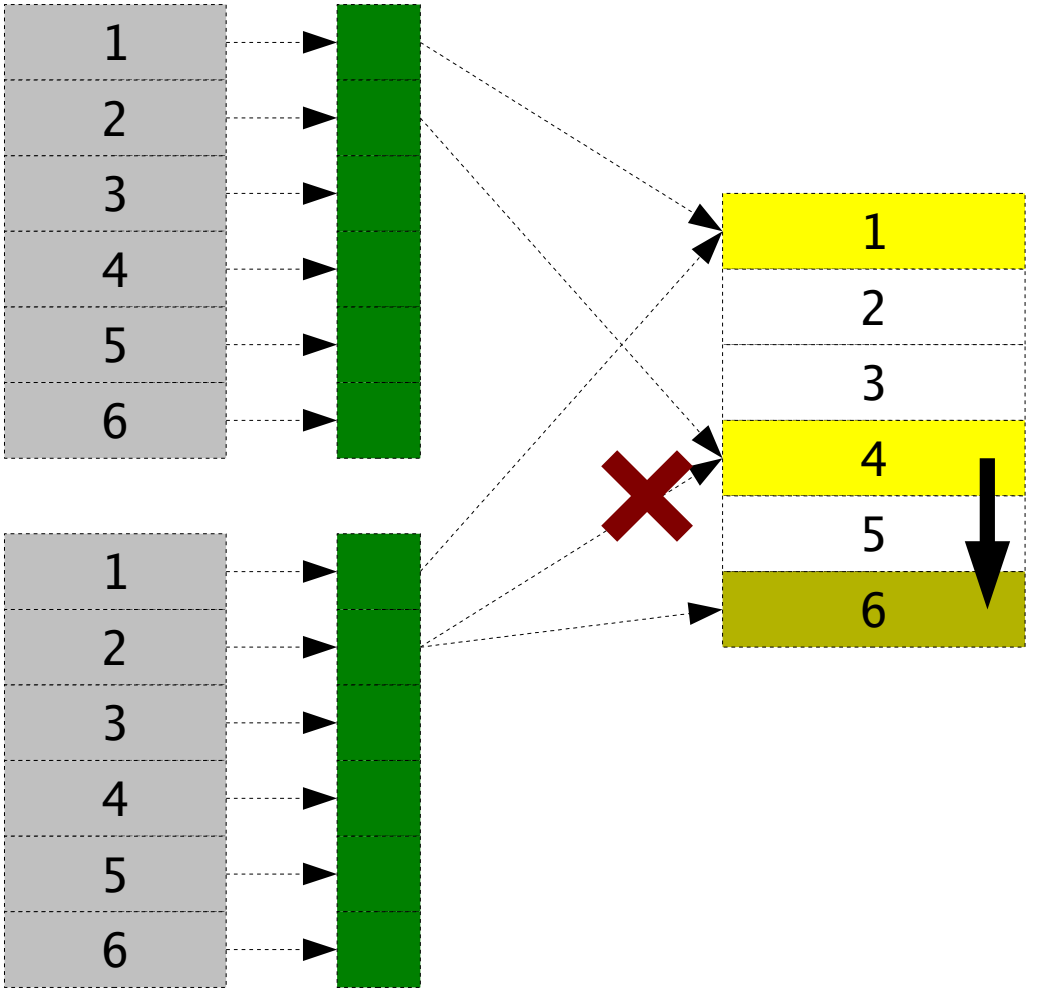
Nowy proces – fork (3)

Specjalnej uwagi wymagają jedynie operacje zapisu na danej stronie pamięci.

Próba modyfikacji danych ze „wspólnej” strony przez jeden z procesów powoduje wstrzymanie procesu.

System operacyjny przydziela zatrzymanemu procesowi dodatkową ramkę i kopiuje zawartość z ramki „wspólnej” aktualizując tablicę stron.

Proces zostaje obudzony i kontynuuje zapis w swoim „prywatnym” obszarze pamięci.



Wymiatanie

- Stronicowanie pozwala na bardzo wydajne zarządzanie pamięcią
- Wielkość pamięci wszystkich procesów razem może znacznie przekroczyć rozmiar pamięci fizycznej.
- Co zrobić gdy jednak zabraknie pamięci fizycznej?

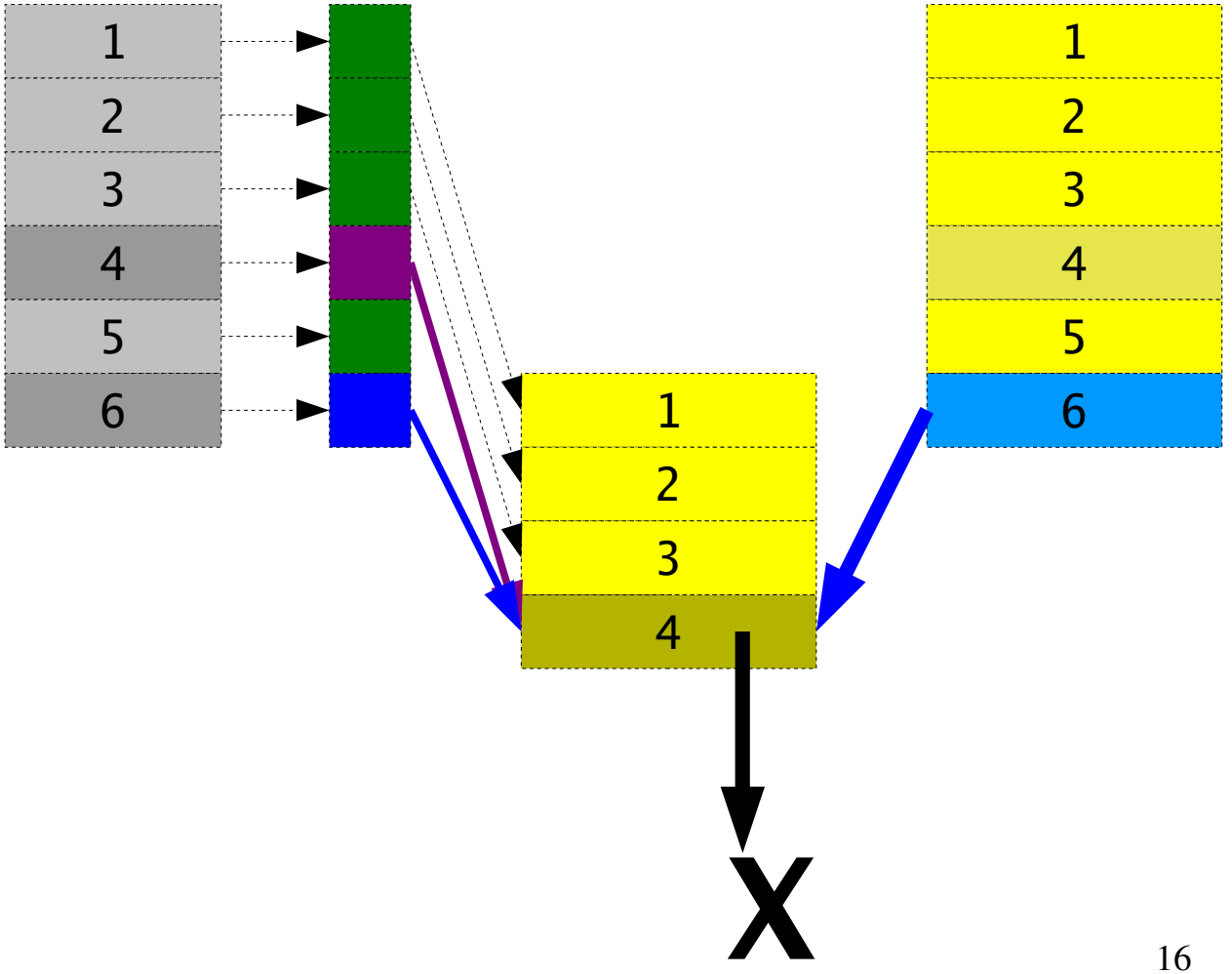
Wymiatanie c.d.

Brak wolnej pamięci operacyjnej.

System może usunąć jedną z niezmodyfikowanych („czystych”) stron jednego z procesów.

Zwolnione miejsce może być wykorzystane do załadowania strony innego procesu

Usunięcie „czystej” strony niczym nie grozi bo dane znajdują się na dysku i mogą być ponownie wczytane.

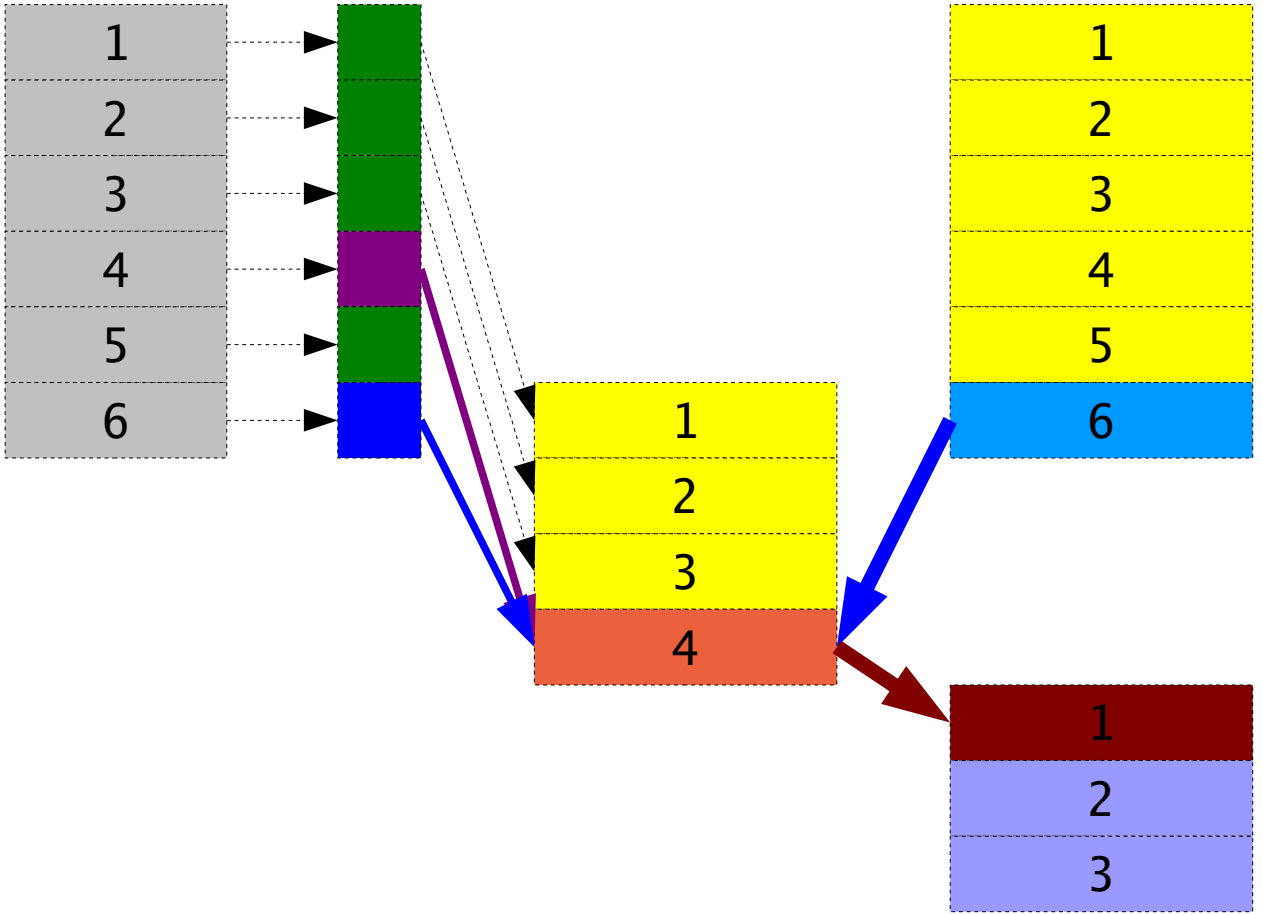


Wymiatanie c.d.

W przypadku braku „czystych” stron do wymiany sprawa się komplikuje.

Należy usunąć stronę zmodyfikowaną („brudną”). Trzeba ją jednak zachować w przypadku ponownego odwołania.

Stronę „brudną” system przenosi na urządzenie wymiany (swap). Jeżeli proces będzie się chciał do niej odwołać to trzeba będzie ponownie umieścić ją w pamięci operacyjnej



Pamięć wirtualna

- Pamięć fizyczna – faktyczna pamięć zainstalowana w systemie
- Pamięć wirtualna – pamięć widziana z punktu widzenia programisty
  - Może znacznie przekraczać rozmiar pamięci fizycznej
  - Ukrywa przed programistą fizyczne ułożenie danych w pamięci fizycznej.

„Migotanie” stron

- Zastosowanie mechanizmu wymiany zwiększa efektywność zarządzania pamięcią i szybkość wykonywania programów
- W pewnych sytuacja może jednak dojść do „migotania” stron – system usuwa strony, do których aplikacja wkrótce się odwoła.
- Efektem „migotania” stron jest dramatyczne zmniejszenie wydajności – system zajmuje się głównie wymianą stron.

Zasada lokalności

- Efektywne wykorzystanie pamięci wirtualnej zakłada „Lokalność czasową i przestrzenną” wykonywanych programów:
  - W pewnym okresie wykonania program będzie wykonywał instrukcje leżące blisko siebie.
  - Uzasadnienie: paradygmaty strukturalny (funkcje) i obiektowy (klasy, metody).

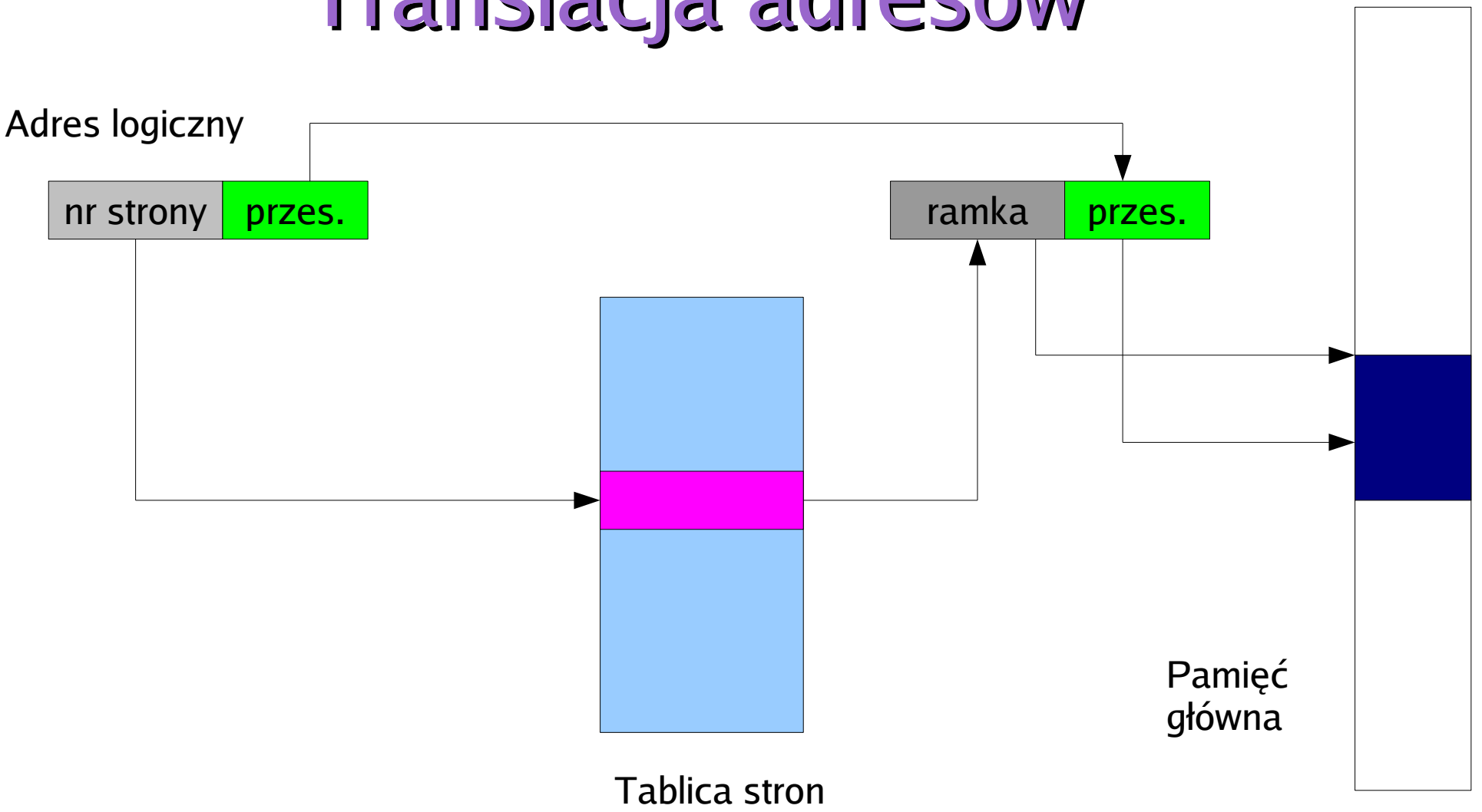
Wsparcie dla pamięci wirtualnej

- Efektywne wykorzystanie pamięci wirtualnej:
  - Wsparcie sprzętu
    - Wsparcie dla stronicowania
    - Efektywne metody wyliczania adresu efektywnego
  - Wsparcie systemu operacyjnego
    - Zarządzanie ładowaniem stron do pamięci
    - Zarządzanie wymianą stron między pamięcią a urządzeniem wymiany

Tablice stron

- Każdy proces ma swoją własną tablicę stron
- Wpis w tablicy stron określa m.in. czy strona znajduje się w pamięci operacyjnej i którą ramkę zajmuje
- Dodatkowo dla każdej strony przechowywany jest bit informujący czy strona była modyfikowana po załadowaniu do pamięci (strona jest „brudna”)

Translacja adresów



Struktura tablicy stron

- Tablica stron procesu może być bardzo duża
- Niektóre systemy pozwalają na przechowywanie tablicy stron w pamięci wirtualnej
- Dwupoziomowa tablica stron
  - Główna tablica stron – zawsze w pamięci operacyjnej – wskazuje na odpowiednią tablicę stron użytkownika
  - Tablice stron – są w pamięci wirtualnej – mogą podlegać mechanizmowi wymiatania

Struktura tablicy stron c.d.

- Odwrócona tablica stron
  - Zastosowanie funkcji mieszającej do określenia pozycji strony w tablicy stron
  - Tablica stron przechowuje zapisy dla poszczególnych ramek pamięci fizycznej
- Bufor translacji adresów (TLB)
  - Przechowuje ostatnio używane elementy tablicy stron
  - Zwiększenie efektywności translacji adresów

Rozmiar strony

- Mały rozmiar strony
  - Mniejsza fragmentacja wewnętrzna
  - Więcej stron – większe tablice stron
  - Lepsze wykorzystanie pamięci (zasada lokalności)
    - mniej błędów braku strony
- Duży rozmiar strony
  - Wydajniejsze transfery (przesłania blokowe)
  - Rozmiar porównywalny z rozmiarem procesu – mniej błędów braku strony

Rozmiar strony c.d.

- Zmienny rozmiar strony – większa elastyczność.
- W praktyce – strony o takim samym rozmiarze
  - VAX 512 bajtów
  - DEC Alpha 8 KB
  - MIPS 4 KB do 16 MB
  - Pentium 4 KB lub 4 MB

Strategia pobierania stron

- Stronicowanie na żądanie (demand paging) – strona umieszczana jest w pamięci operacyjnej tylko jeżeli pojawia się do niej odwołanie. Zwykle powoduje dużo błędów braku stron na początku wykonania procesu.

- Wstępne stronicowanie (prepaging) – pobierane jest kilka stron na raz – wykorzystywanie przesłań blokowych pamięci masowych.

Strategie rozmieszczania stron

- W przypadku systemów ze stronicowaniem lub stronicowaniem i segmentacją rozmieszczanie ramek w pamięci nie ma dużego znaczenia

- Wyjątkiem może być architektura NUMA (niejednolity system dostępu do pamięci w systemach wieloprocesorowych).

Blokowanie ramek

- Niektóre ramki mogą być “zablokowane” w pamięci tzn. nie są one wymieniane.
- Przykłady:
  - Część jądra systemu
  - Struktury danych jądra systemu
  - Bufory urządzeń
  - Obszary krytyczne pod względem czasu dostępu

Strategie wymiany

- Określa które strony są dobrymi kandydatami do wymiany.
  - LRU (Least Recently Used) – strategia najdłużej ostatnio nieużywanej strony.
  - Cykliczny bufor FIFO – pierwsza na wejściu, pierwsza na wyjściu
  - Różne warianty algorytmu zegara

Strategia LRU

- Wymieniona zostaje strona, która najdłużej nie była używana.
- Zakłada regułę „lokalności czasowej i przestrzennej programu”.
- Wszystkie strony muszą zawierać informację o czasie ostatniego (dostępu) – duży narzut.

Strategia FIFO

- Strategia wymiany na zasadzie FIFO (First-In, First-Out).
- Prosta obsługa – bufor kołowy stron procesu.
- Usuwana jest strona, która była w pamięci najdłużej.
- Sprzeczne z zasadą lokalności – może powodować „migotanie” stron.

Algorytm zegarowy

- Bufor kołowy stron do usunięcia.
- Dodatkowy bit (bit wykorzystania)
  - Załadowanie strony – bit ustawiany na 1
  - Odwołanie do strony – bit ustawiany na 1
  - Poszukiwanie kandydata do wymiany
    - Przestawienie bitu wykorzystania z 1 na 0 lub
    - Usunięcie strony

Zestaw rezydentny stron i strategie przydziału

- Zestaw rezydentny – zestaw stron danego procesu przechowywany w pamięci.
- Strategie przydziału:
  - Stały przydział – stała liczba ramek, ustalana przy starcie procesu
  - Zmienny przydział – zmiana liczby ramek przypisanych procesowi w trakcie jego działania.

Zakres wymiany

- Strategie zakresu wymiany:
  - Lokalna – wymianie podlegają tylko rezydentne strony procesu który wywołał błąd strony
  - Globalna – kandydatami do wymiany mogą być wszystkie niezablokowane strony w systemie

Segmentacja

- W odróżnieniu od stron segmenty mogą być różnej wielkości
- Ułatwia tworzenie programów (moduły)
- Wsparcie dla mechanizmów ochrony – segmenty o określonych funkcjach (kod, dane ...)
- SO może modyfikować rozmiar segmentu w czasie wykonania programu.

Segmentacja

- Adres efektywny – numer segmentu, przesunięcie
- Tablica segmentów – przechowuje adres początku segmentu i jego rozmiar.
- Translacja adresów jest analogiczna jak w przypadku stronicowania
- Dodatkowe informacje w tablicy segmentów: segment w pamięci, segment modyfikowany, flagi (wykonywalny, tylko do odczytu itd.)

Stronicowanie i segmentacja

- Współczesne systemy operacyjne często realizują pamięć wirtualną jako połączenie stronicowania i segmentacji
- Stronicowanie – niewidoczne dla programisty, lepsze zarządzanie pamięcią
- Segmentacja – kontrolowana przez programistę, elastyczne struktury danych, modularność programu, ochrona.