Efektywny mechanizm copy-on-write dla systemu operacyjnego Mimiker

Franciszek Zdobylak

Instytut Informatyki
Uniwersytet Wrocławski

22 marca 2024

Spis treści

1 Pamięć wirtualna

2 Pamięć wirtualna w systemie NetBSD (oraz Mimikerze)

3 Implementacja CoW w Mimikerze

4 Podsumowanie

Pamięć wirtualna

Pamięć fizyczna

Budowa

- Główna pamięć – RAM
  - Jednolita
  - Dostęp z wykorzystaniem adresów (fizycznych)
- Urządzenia pamięci
  - Dyski, Pamięć flash itp.
- Inne urządzenia
  - Karta graficzna, karta dźwiękowa itp.

Pamięć wirtualna

Warstwa abstrakcji nad pamięcią fizyczną.

Zadania:

Pamięć wirtualna

Warstwa abstrakcji nad pamięcią fizyczną.

Zadania:

1. Zapewnienie bezpieczeństwa
   - izolacja i kontrola dostępu

Pamięć wirtualna

Warstwa abstrakcji nad pamięcią fizyczną.

Zadania:

1. Zapewnienie bezpieczeństwa
   - izolacja i kontrola dostępu
2. Efektywne wykorzystanie pamięci RAM

Pamięć wirtualna

Warstwa abstrakcji nad pamięcią fizyczną.

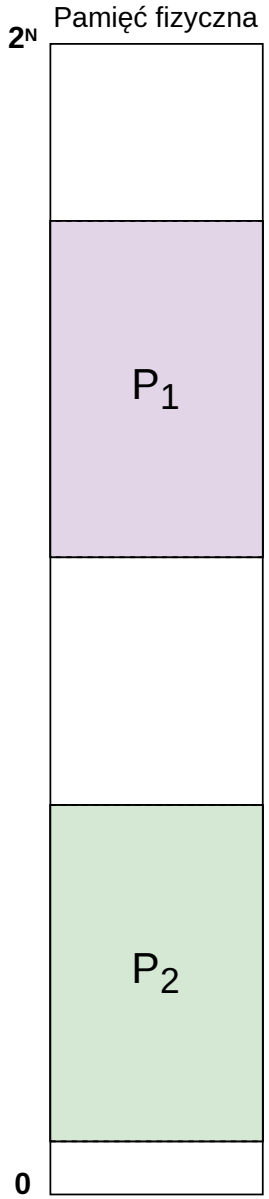
Zadania:

1. Zapewnienie bezpieczeństwa
   - izolacja i kontrola dostępu
2. Efektywne wykorzystanie pamięci RAM
3. Uproszczenie widoku pamięci przez procesy
   - iluzja posiadania całej pamięci dla siebie

Mechanizmy pamięci wirtualnej

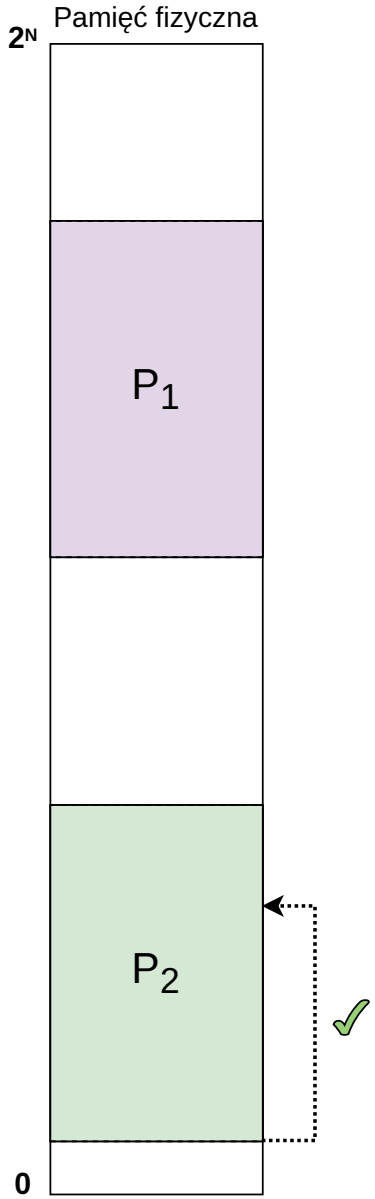
Adresy fizyczne vs wirtualne

Izolacja przestrzeni adresowych



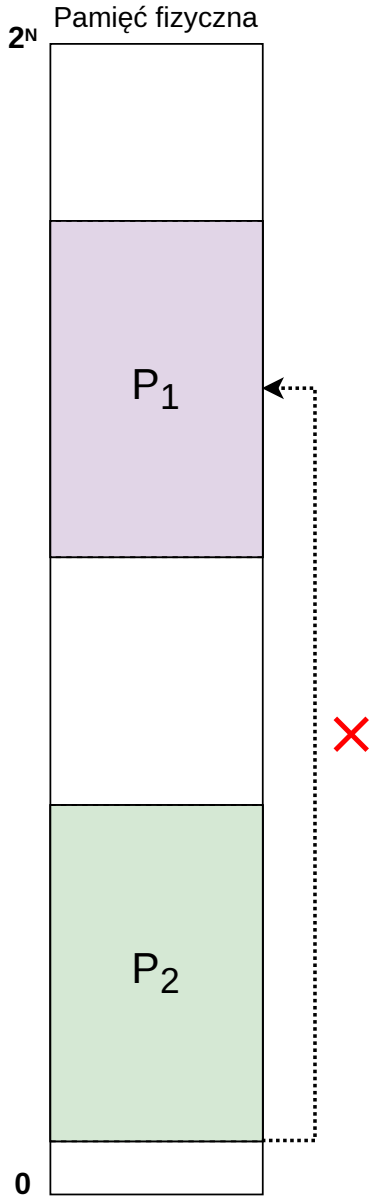
Adresy fizyczne vs wirtualne

Izolacja przestrzeni adresowych



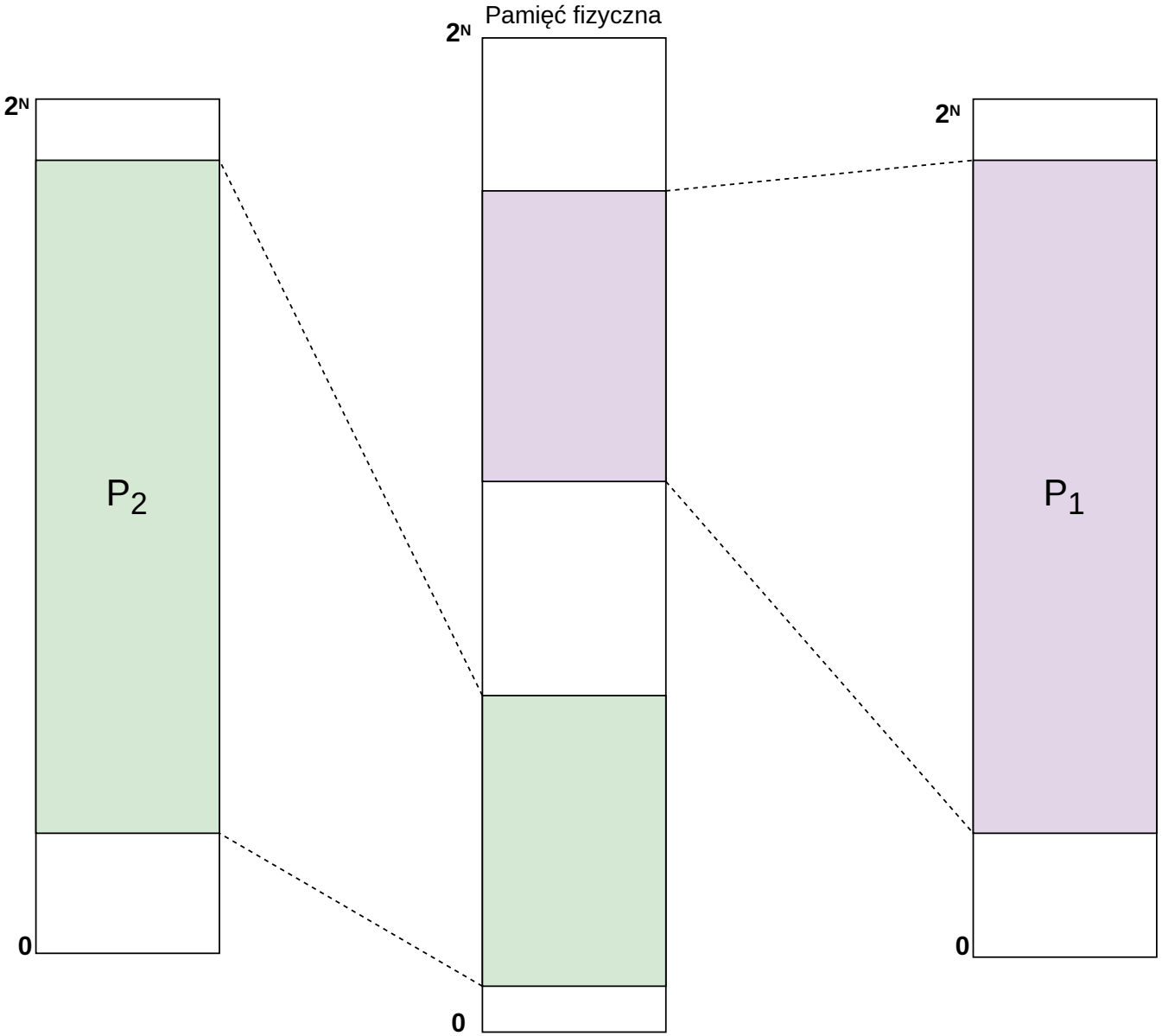
Adresy fizyczne vs wirtualne

Izolacja przestrzeni adresowych



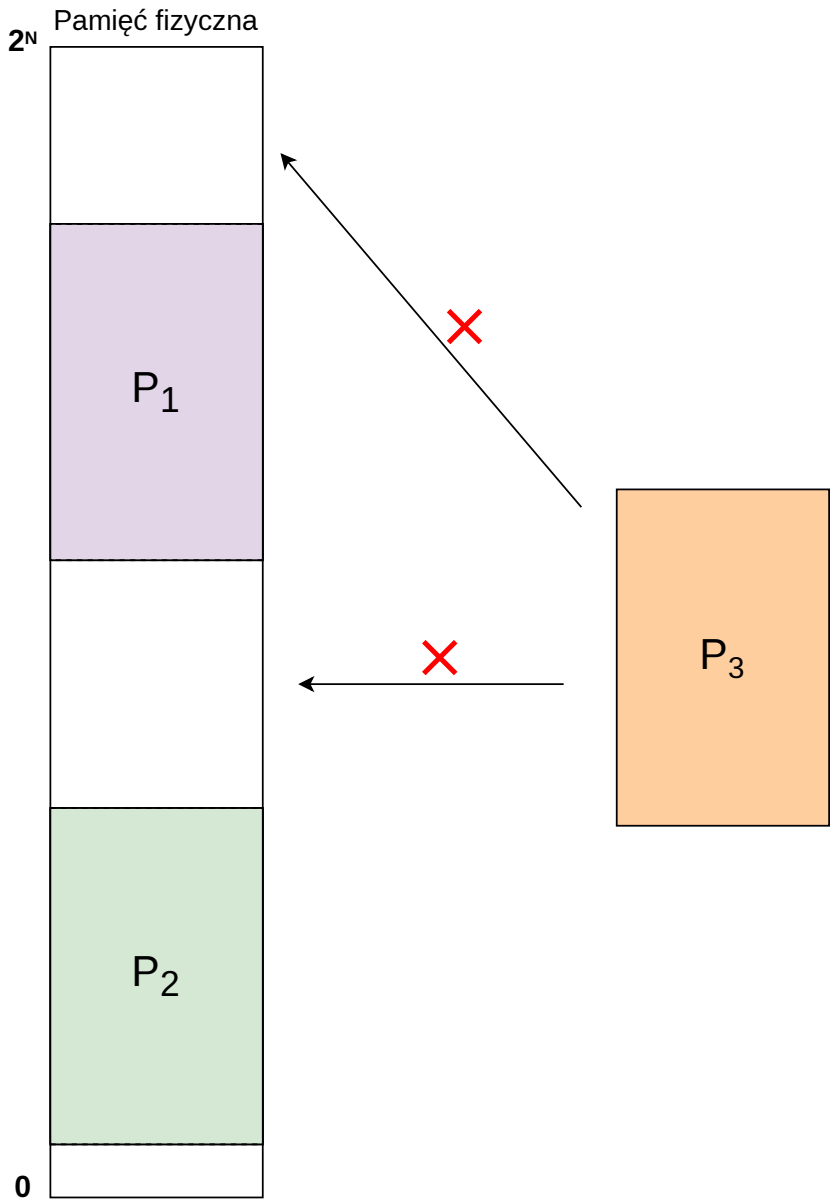
Adresy fizyczne vs wirtualne

Izolacja przestrzeni adresowych



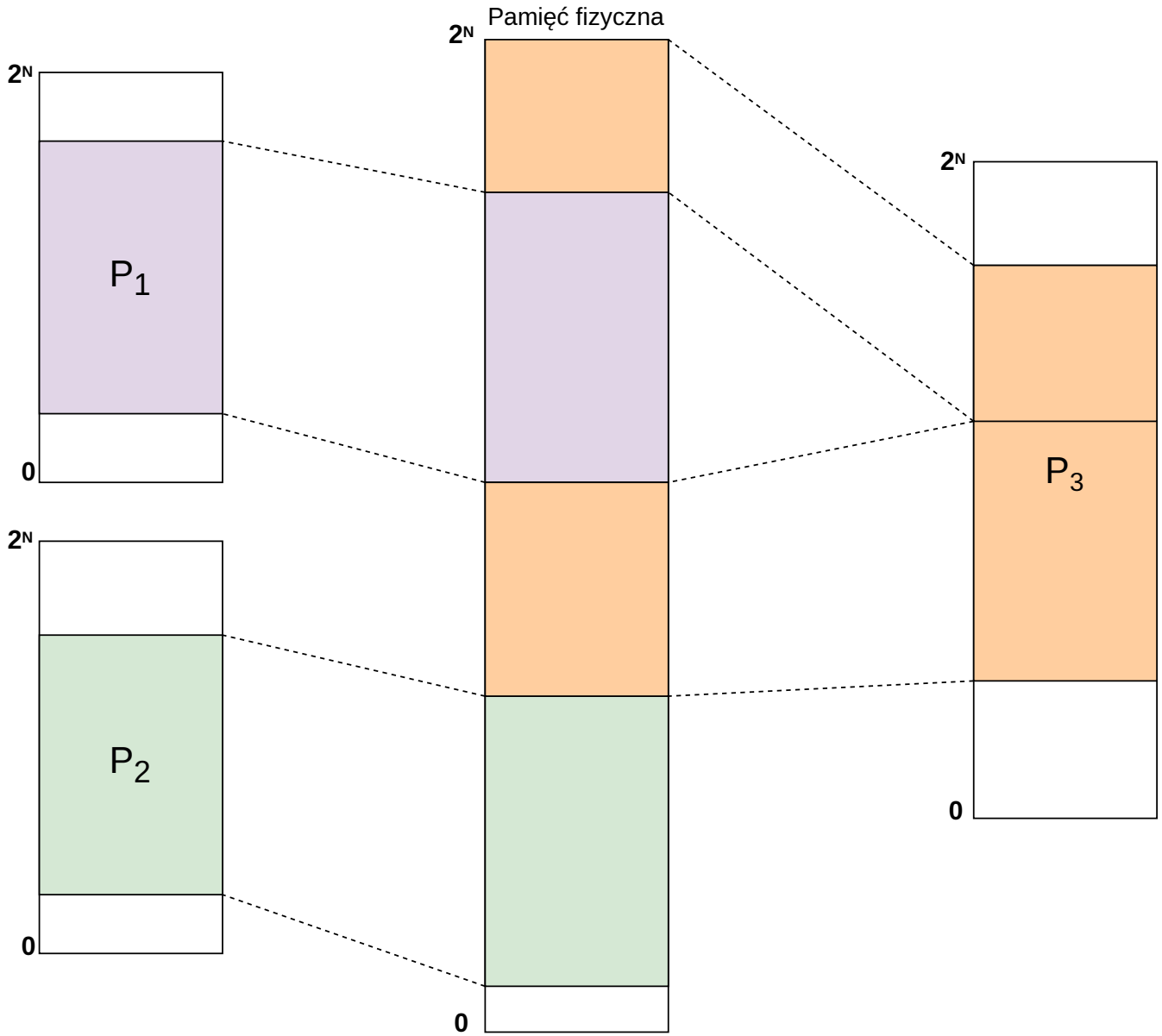
Adresy fizyczne vs wirtualne

Efektywne wykorzystanie pamięci

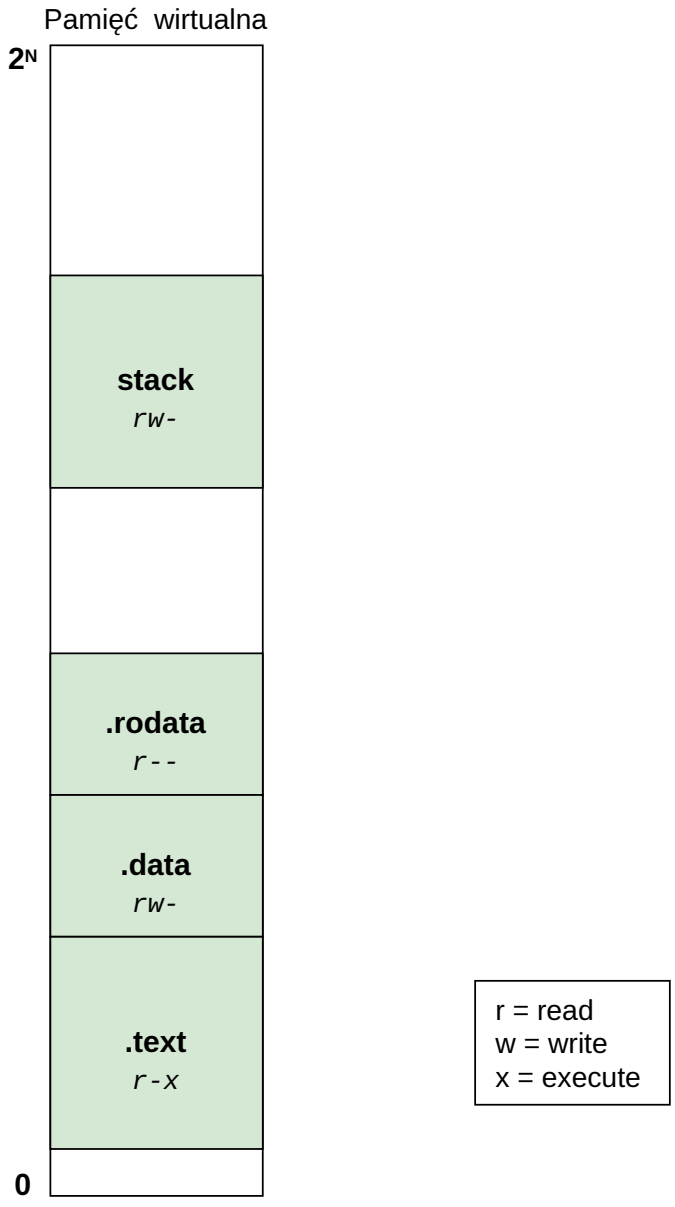


Adresy fizyczne vs wirtualne

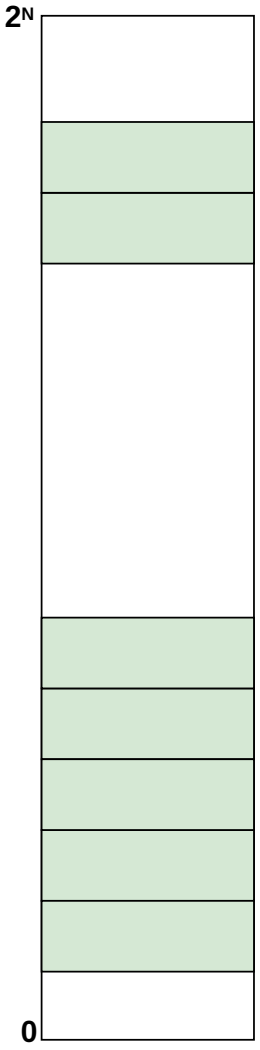
Efektywne wykorzystanie pamięci



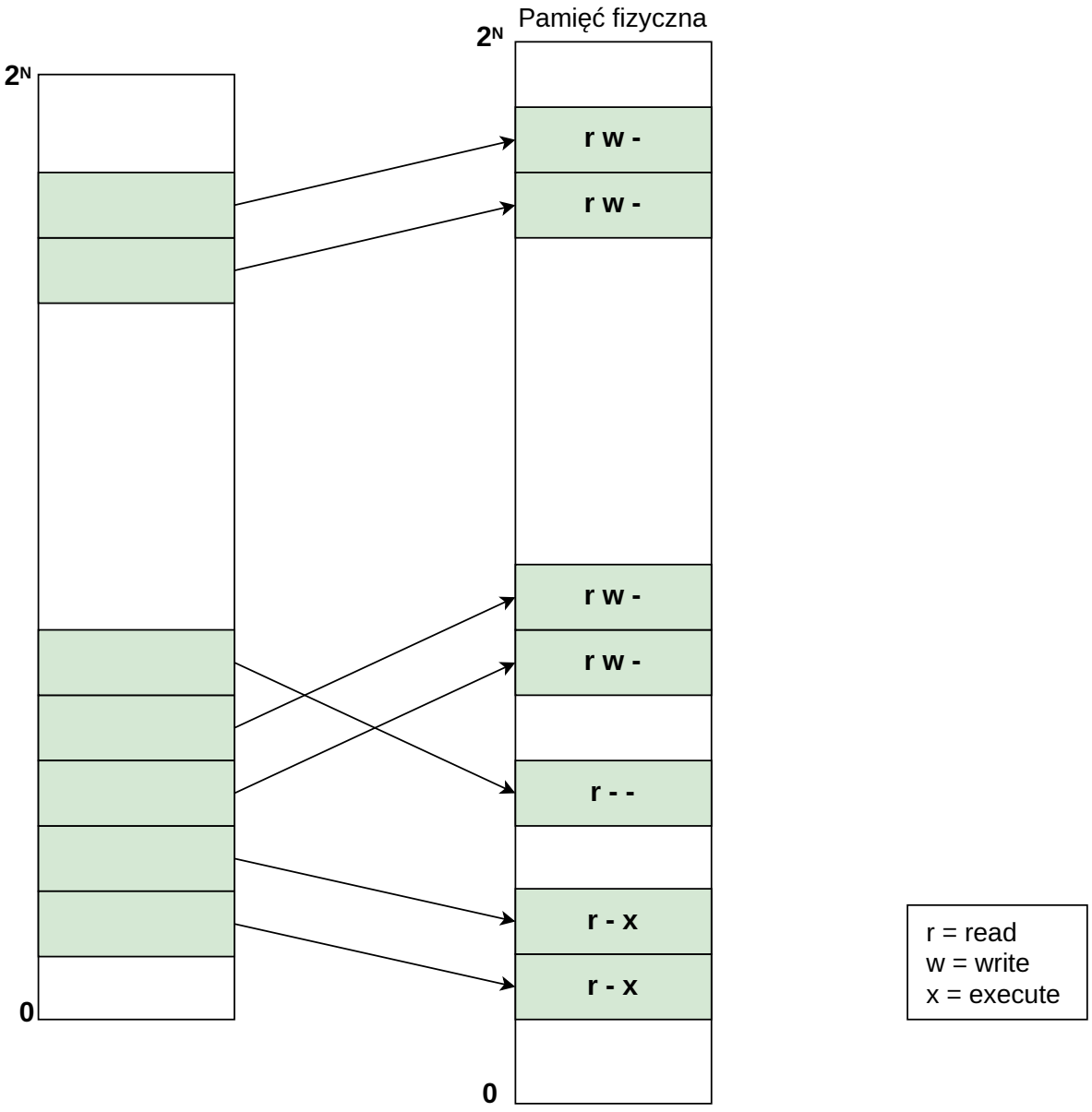
Jednakowy widok pamięci



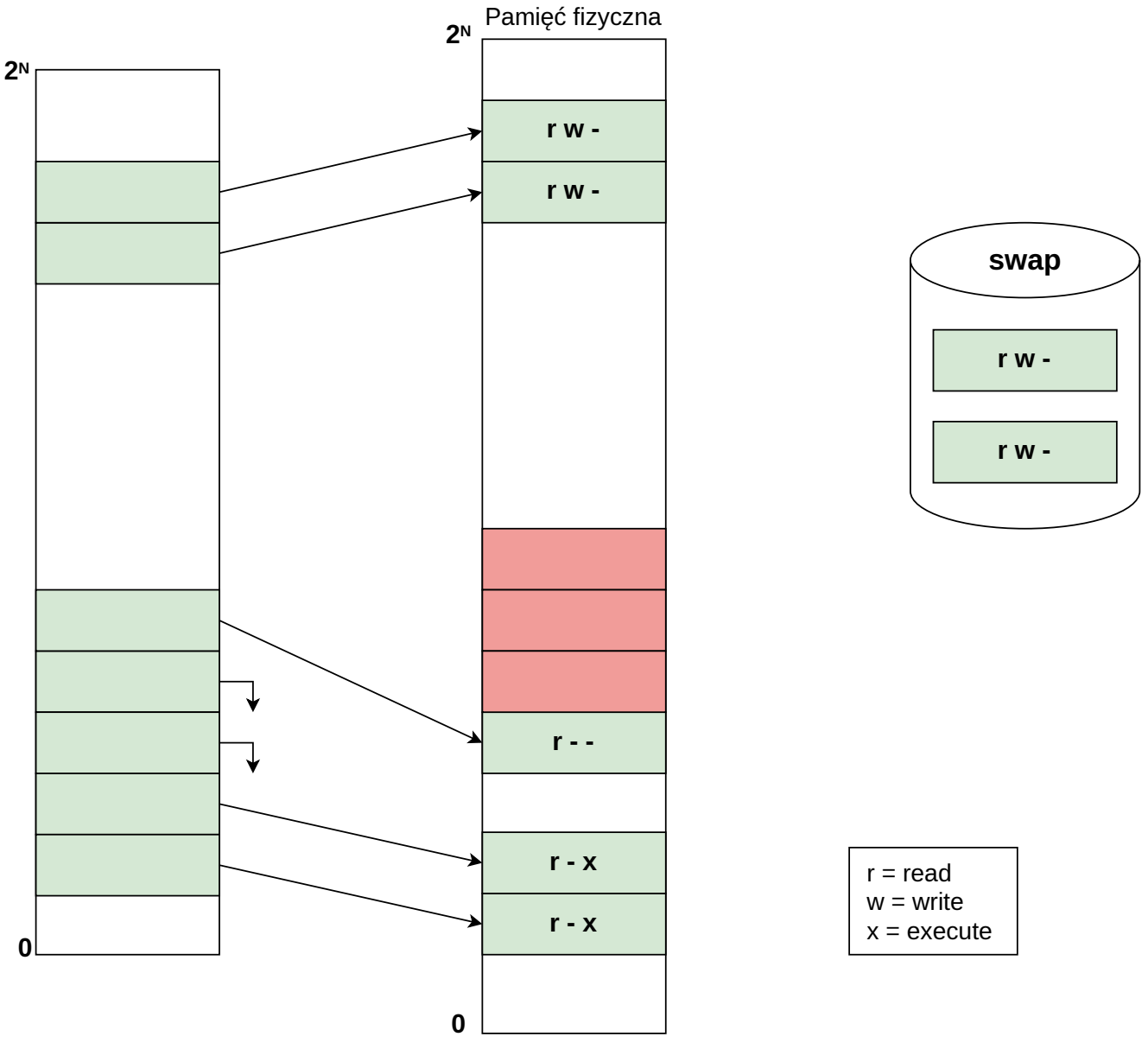
Stronicowanie



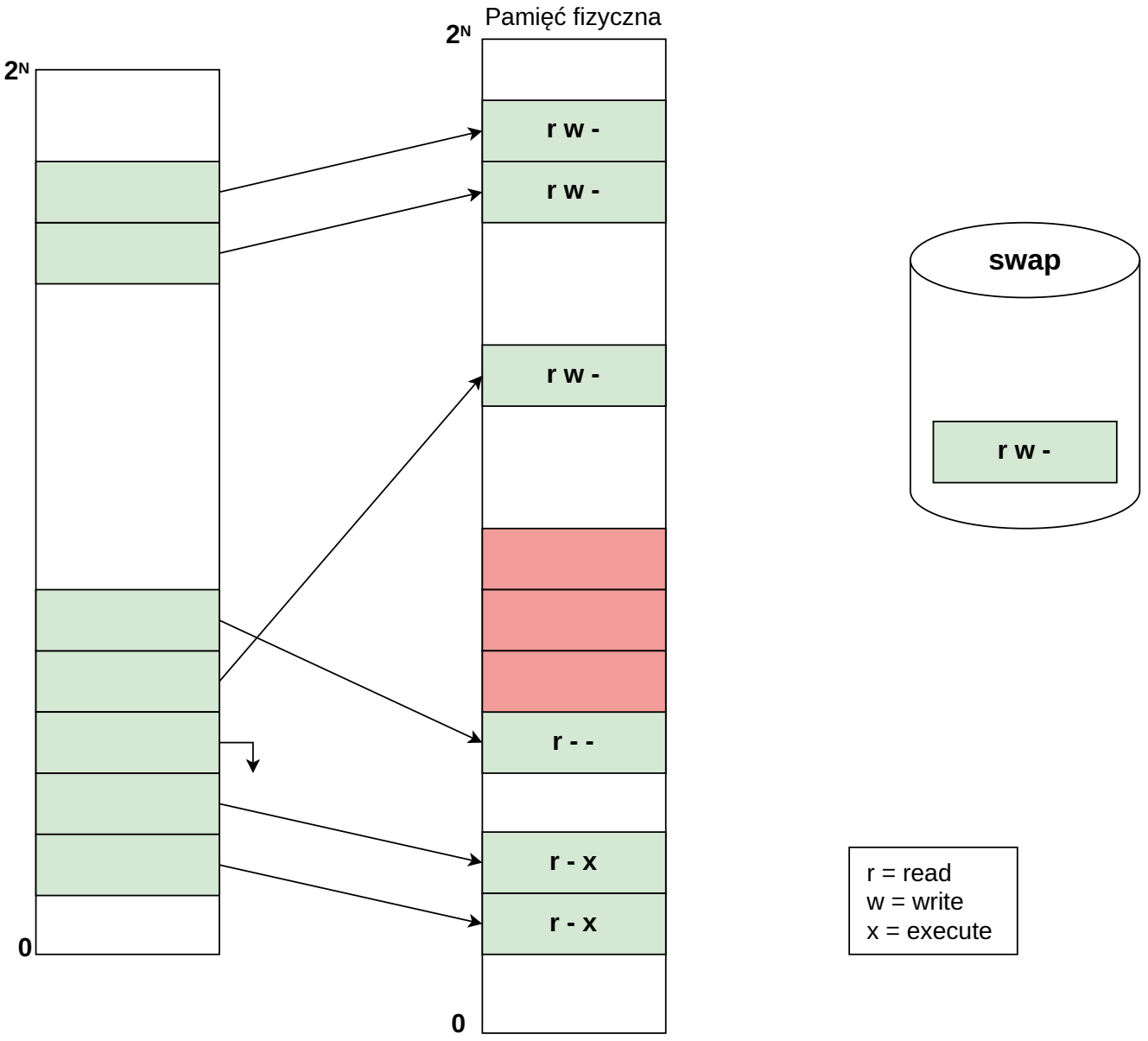
Stronicowanie



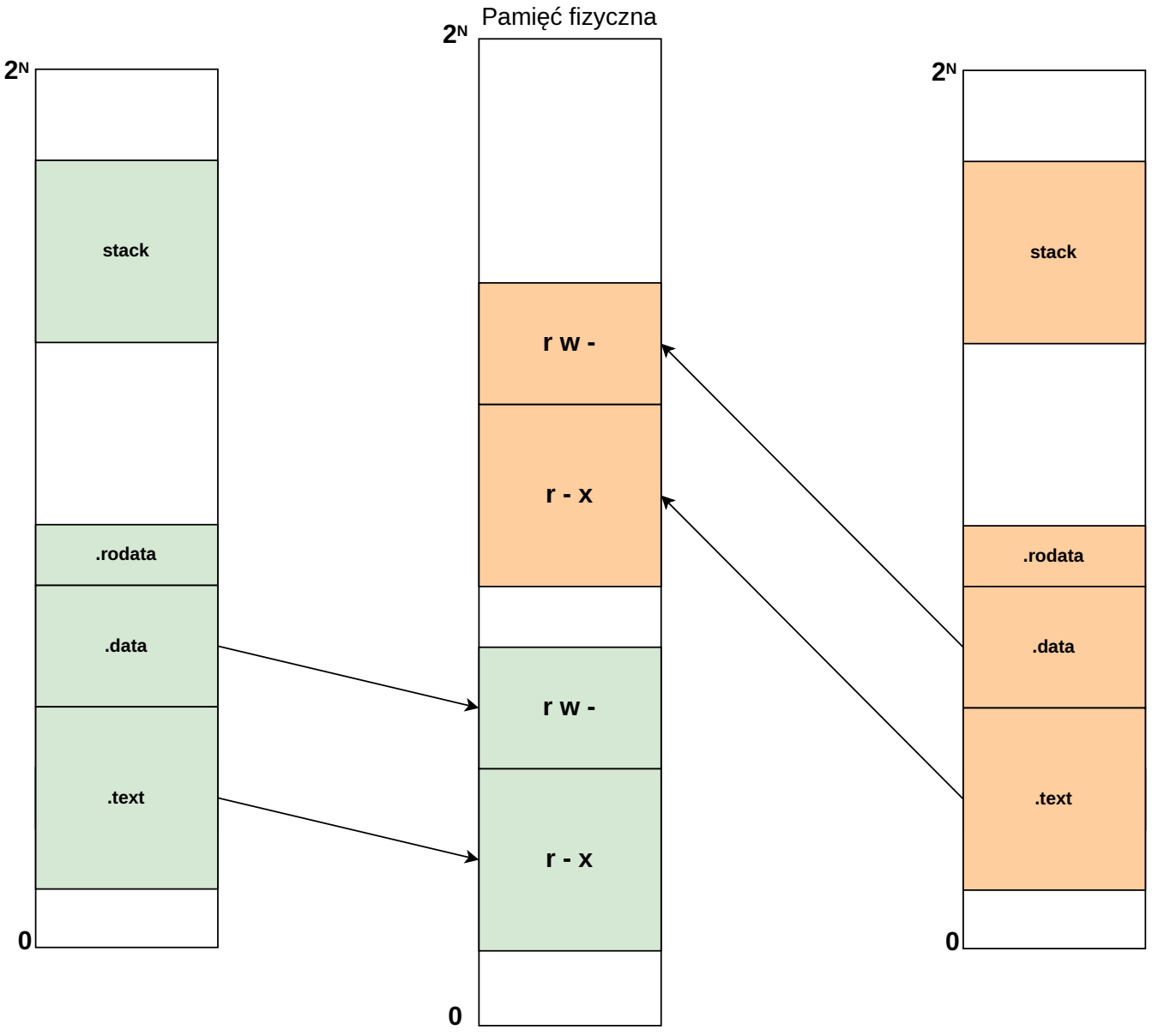
Wymiana stron



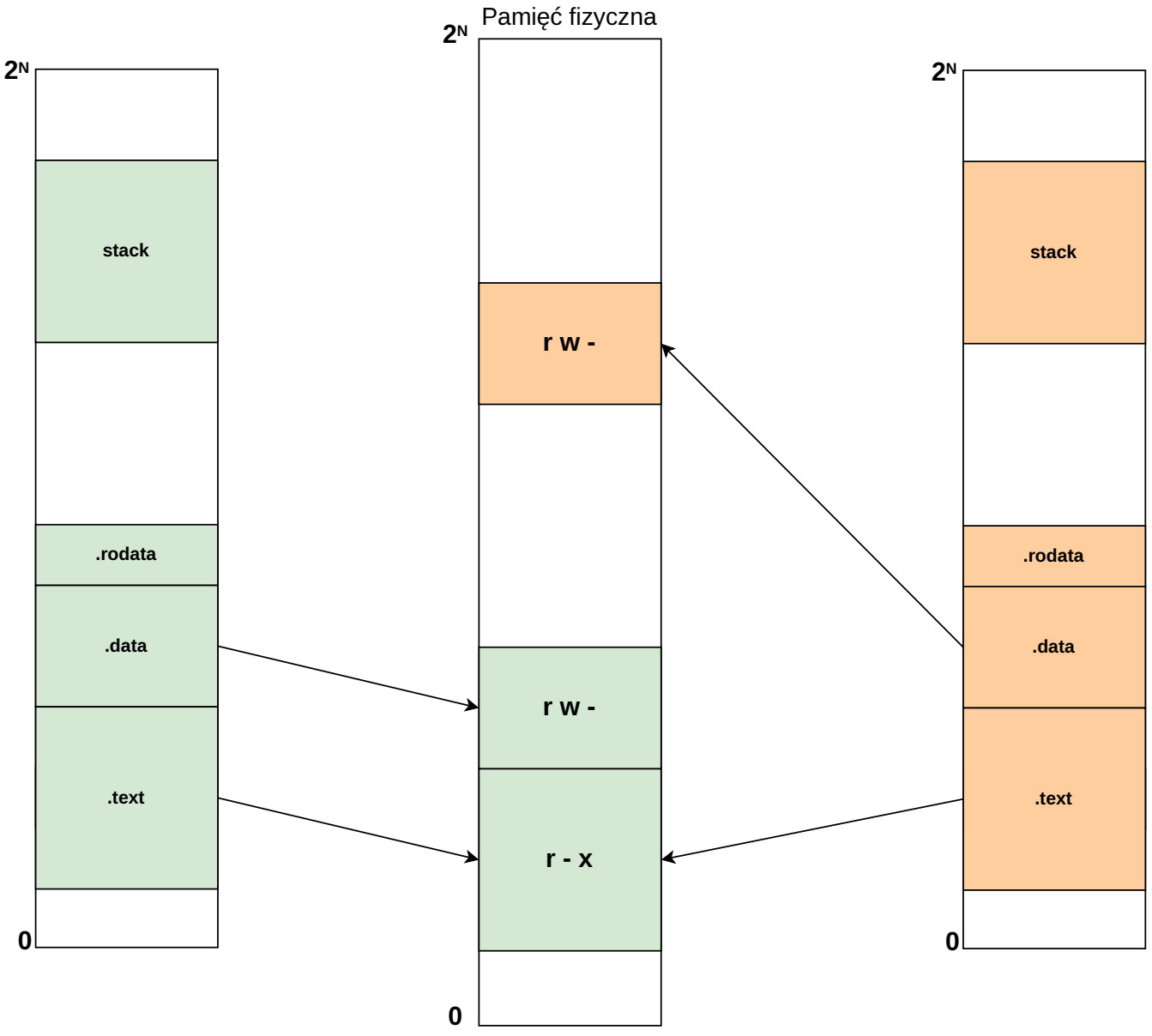
Wymiana stron



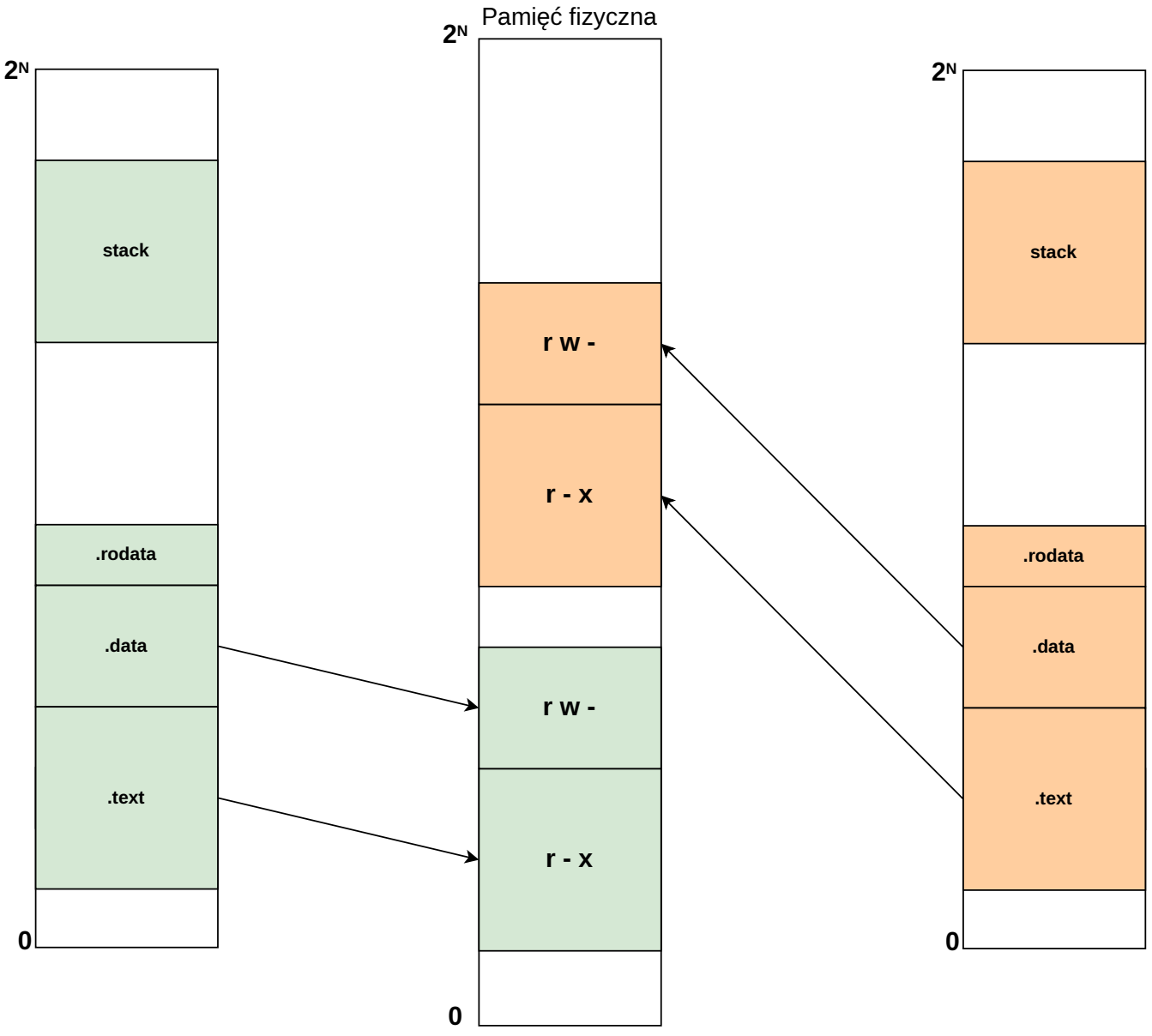
Współdzielenie pamięci



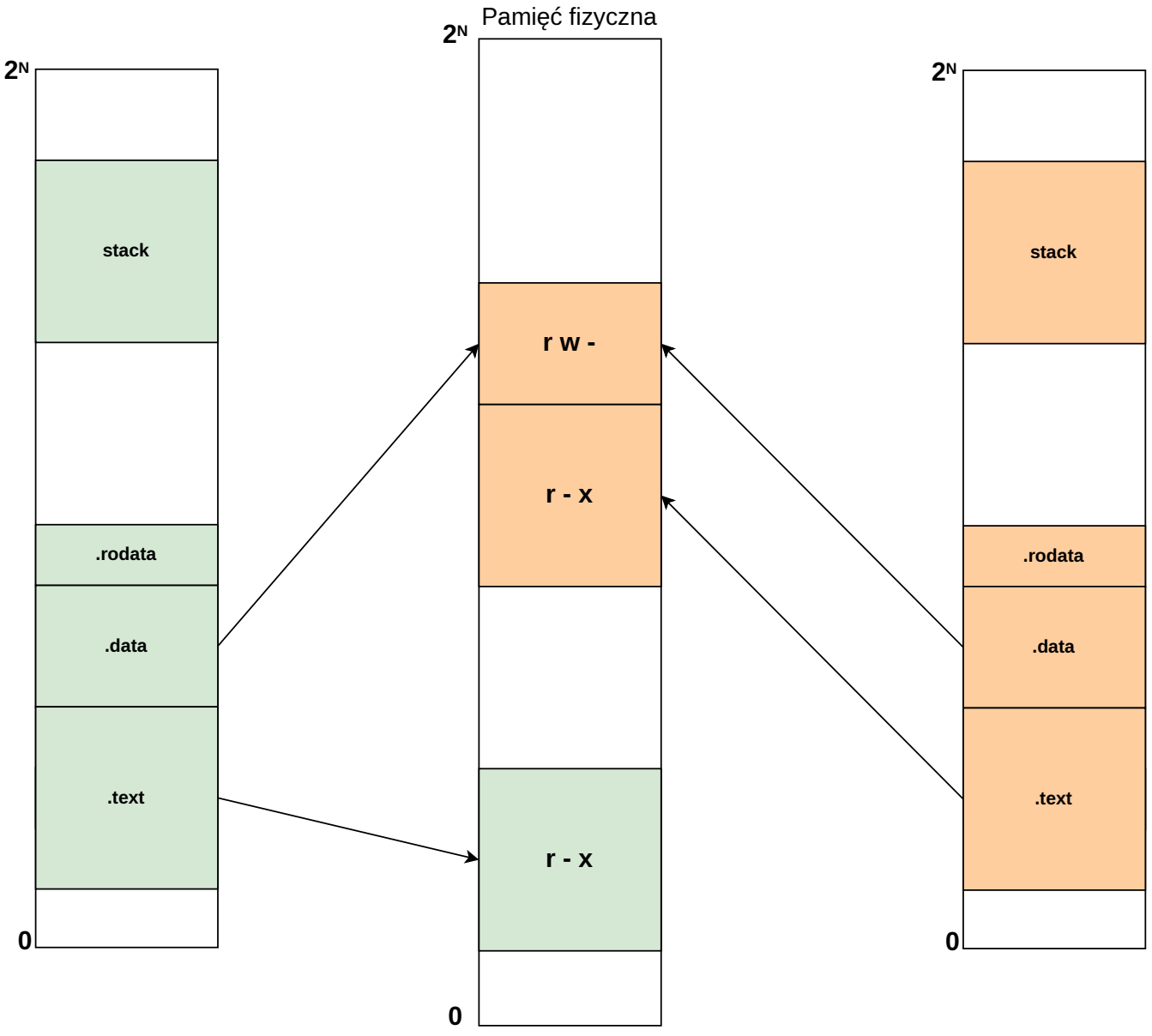
Współdzielenie pamięci



Współdzielenie pamięci



Współdzielenie pamięci



Mechanizm copy-on-write (CoW)

Optymalizacja wywołania `fork`.

Mechanizm copy-on-write (CoW)

Optymalizacja wywołania `fork`.

Fork bez CoW:

- Kopiuje całą przestrzeń adresową
- Rodzic oraz dziecko posiadają prywatne kopie pamięci

Mechanizm copy-on-write (CoW)

Optymalizacja wywołania `fork`.

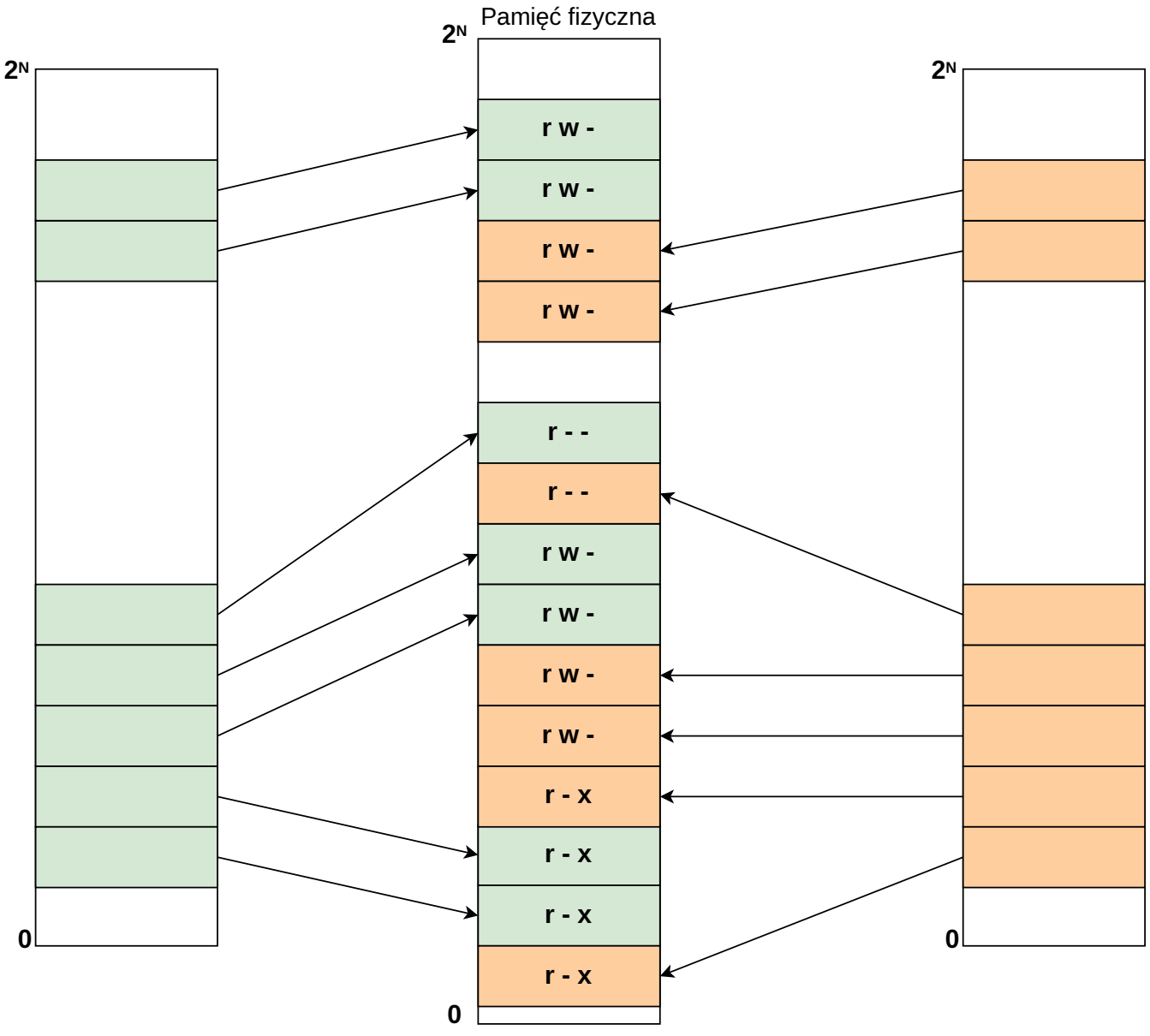
Fork bez CoW:

- Kopiuje całą przestrzeń adresową
- Rodzic oraz dziecko posiadają prywatne kopie pamięci

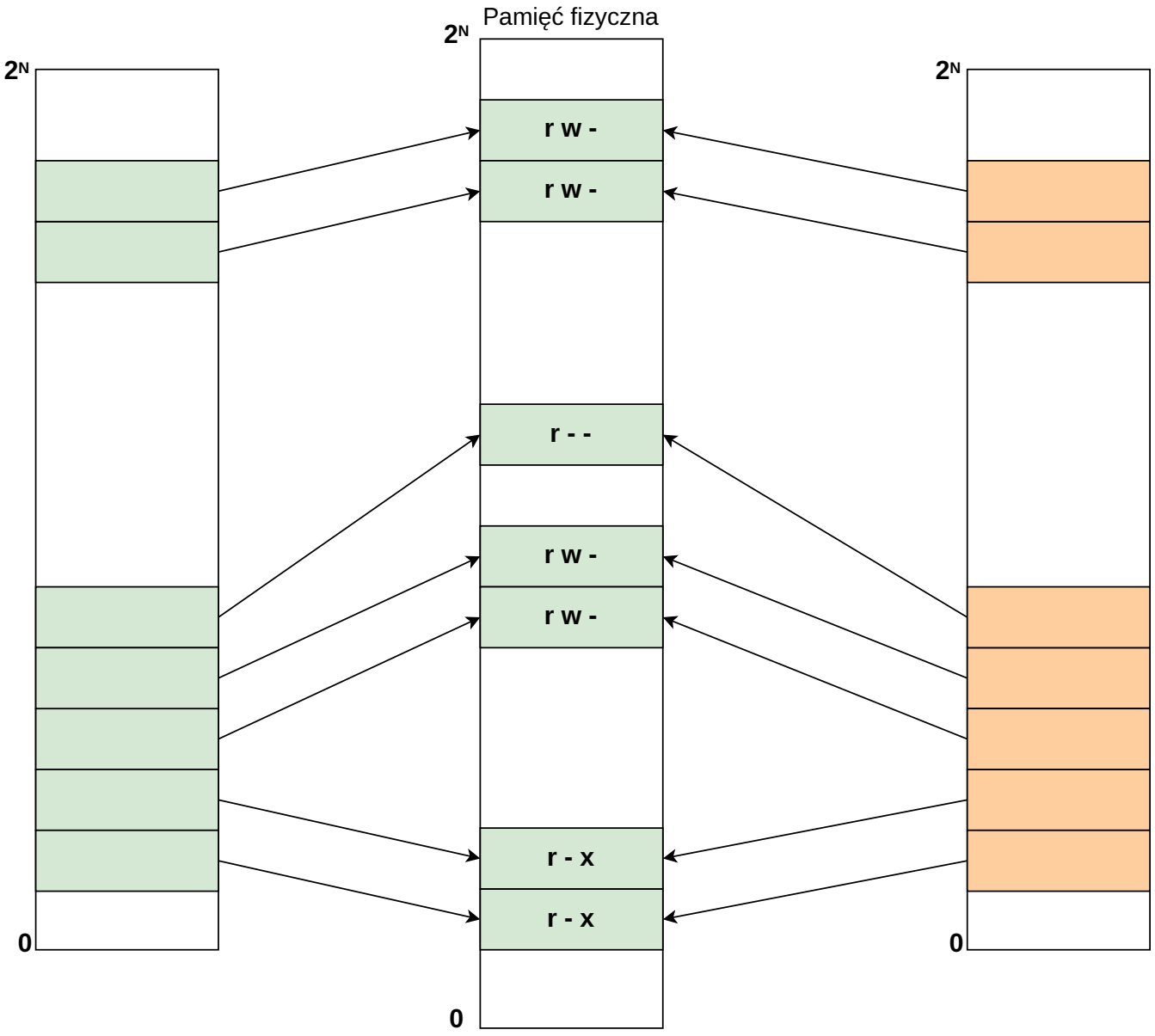
Fork + CoW:

- Nie kopiuje pamięci
- Oznacza pamięć jako "CoW"
- Kopiuje pojedyncze strony przy pierwszym zapisie

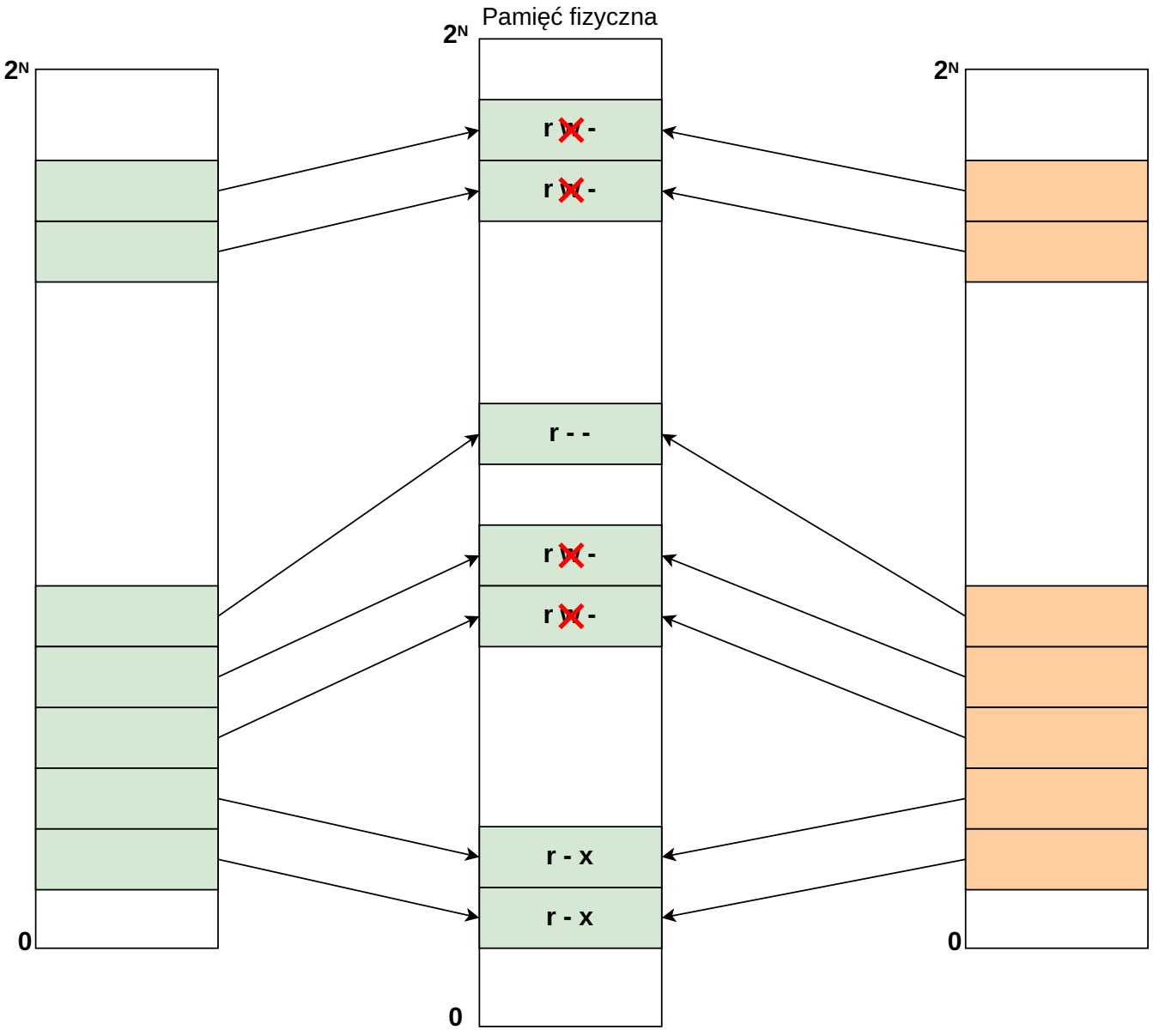
Standardowy `fork`



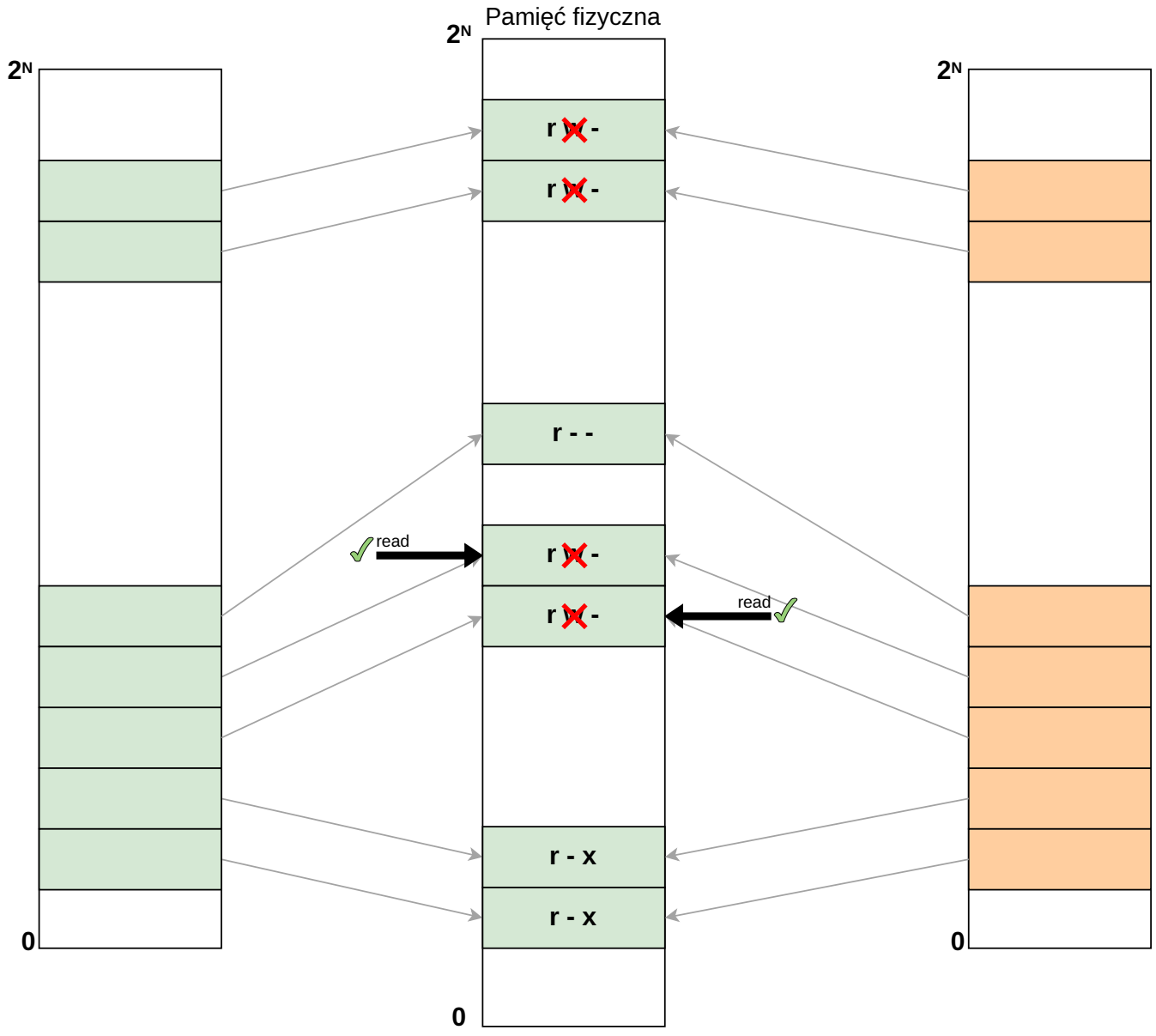
fork + CoW



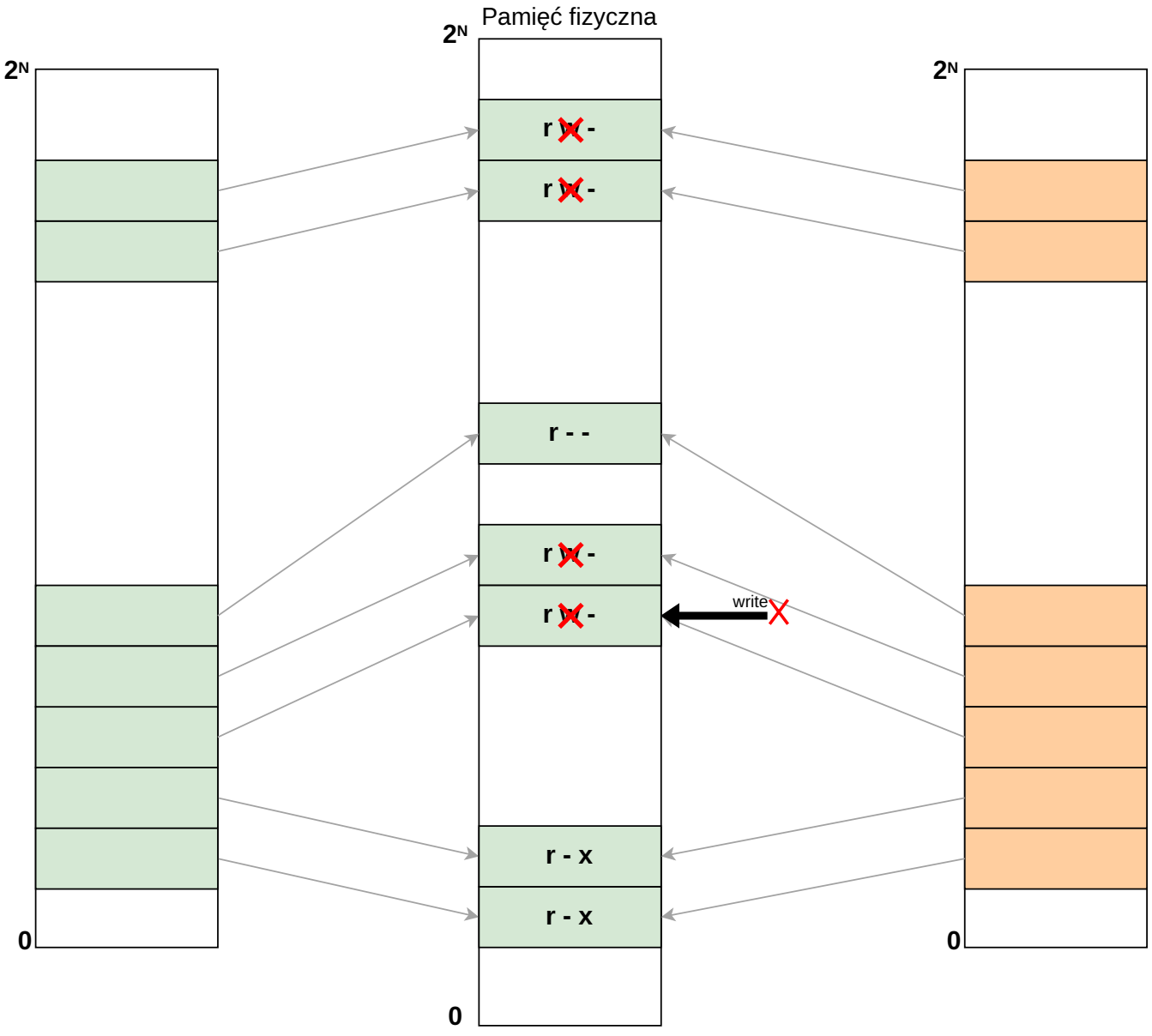
fork + CoW



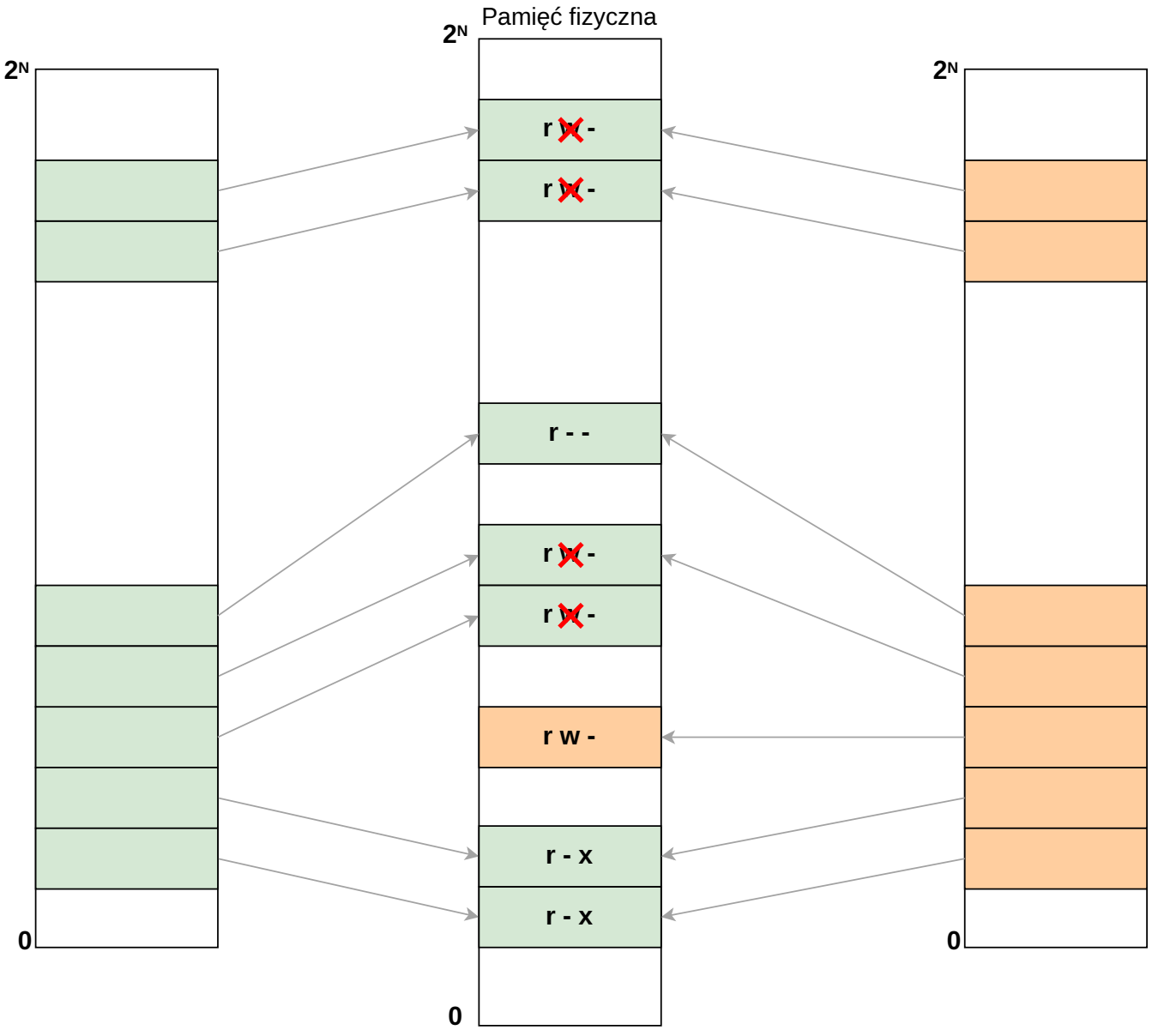
Copy-on-write (dostępy do pamięci)



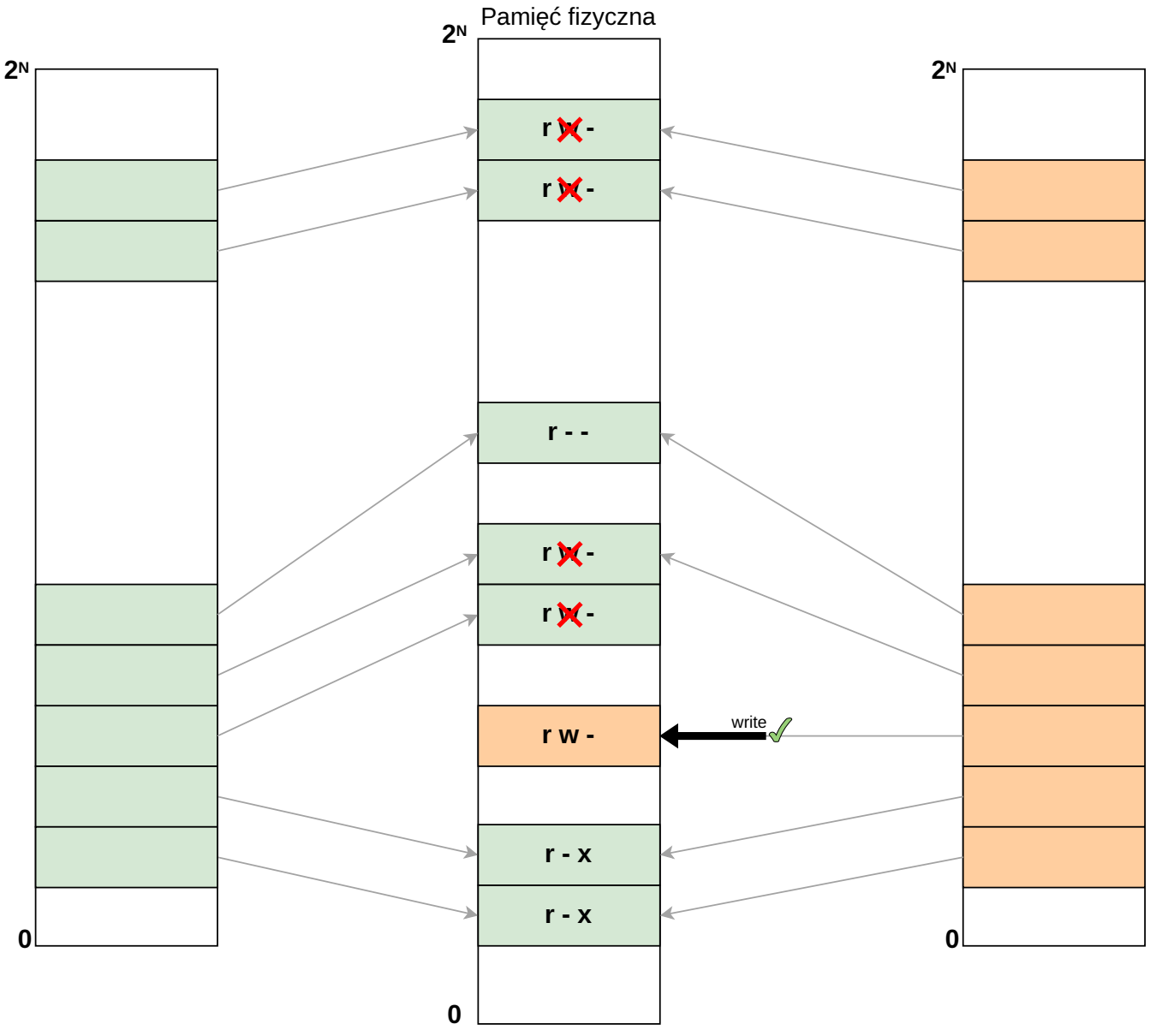
Copy-on-write (page fault)



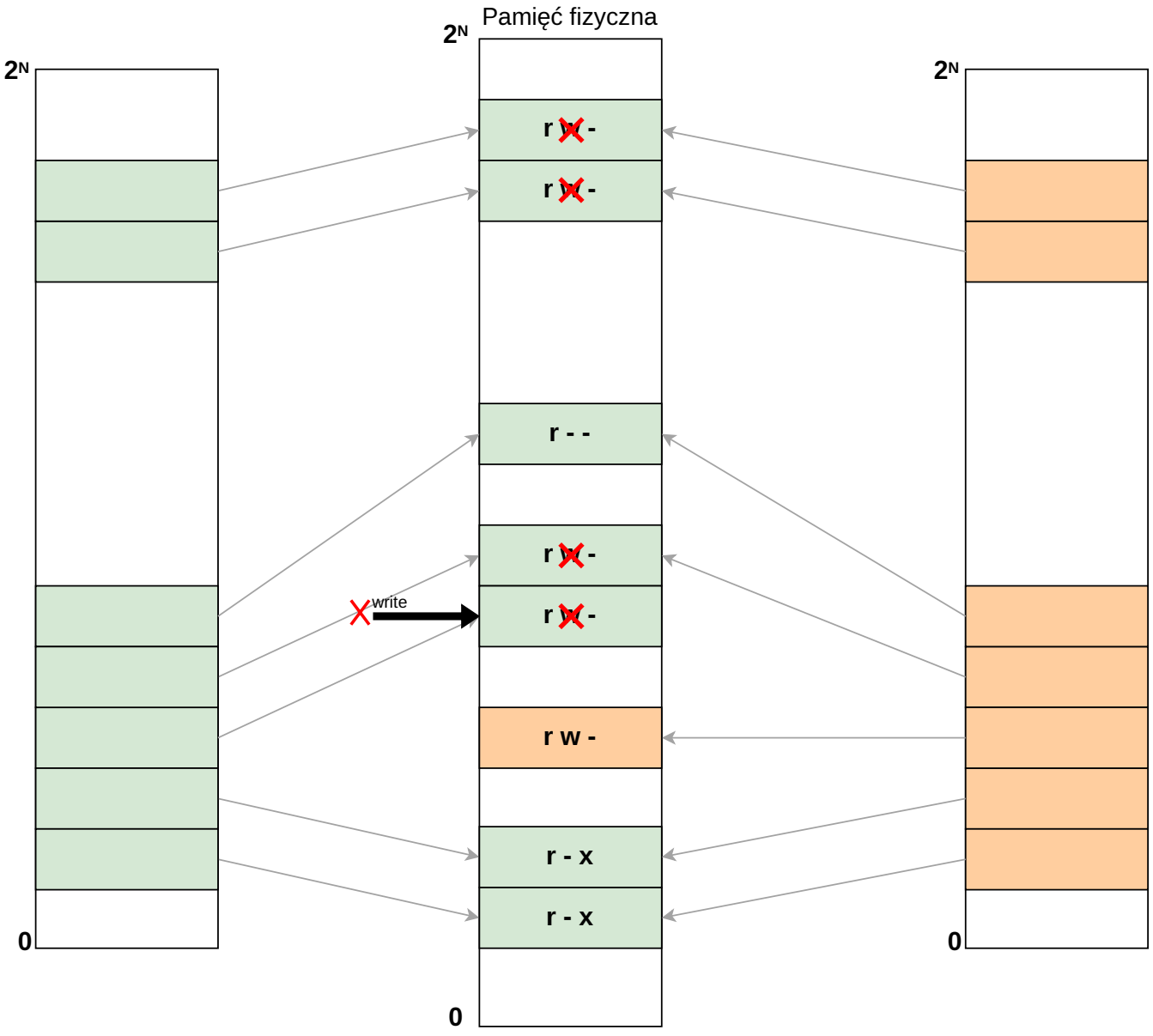
Copy-on-write (kopia)



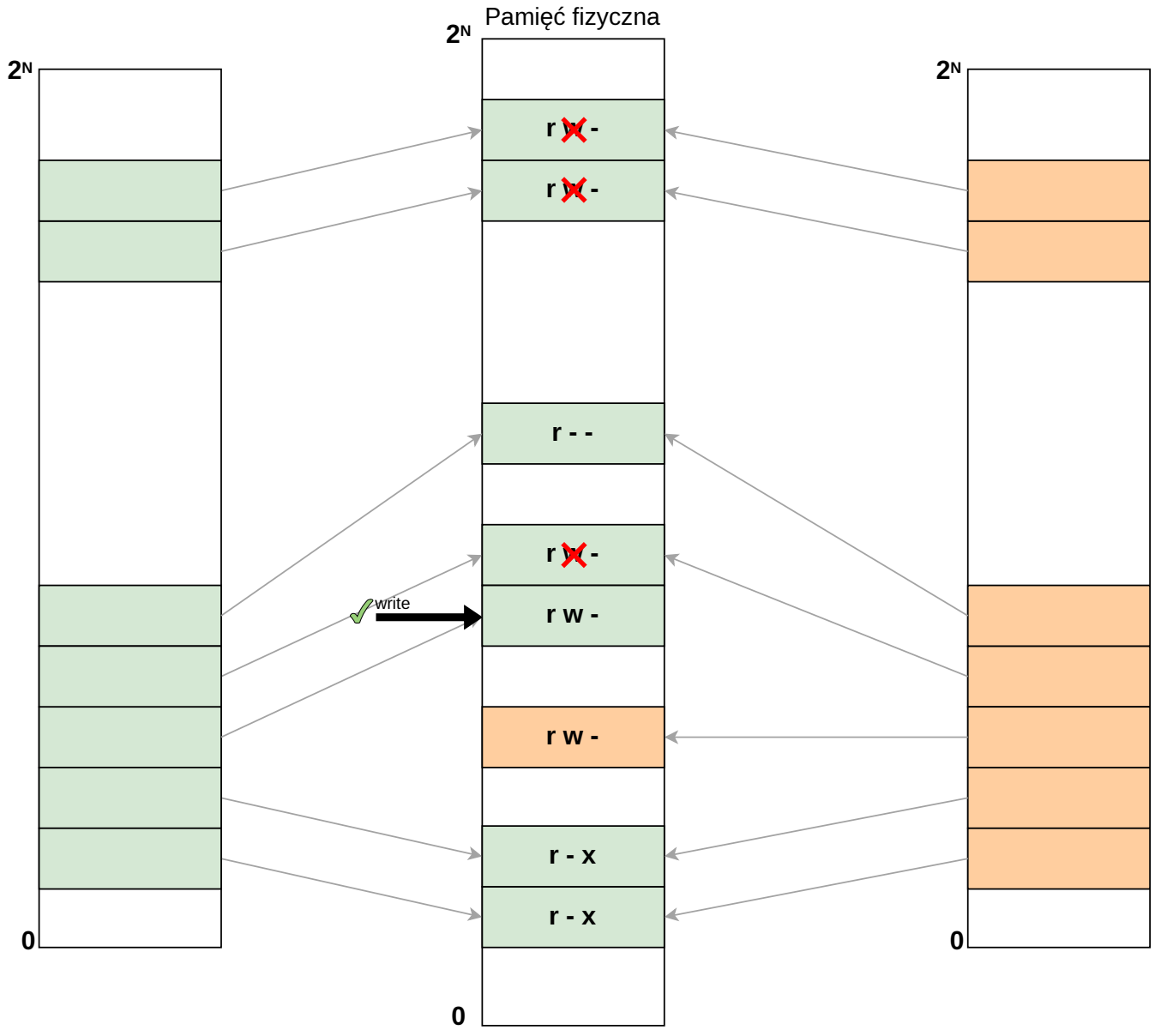
Copy-on-write (zapisy do pamięci)



Copy-on-write (zapisy do pamięci)



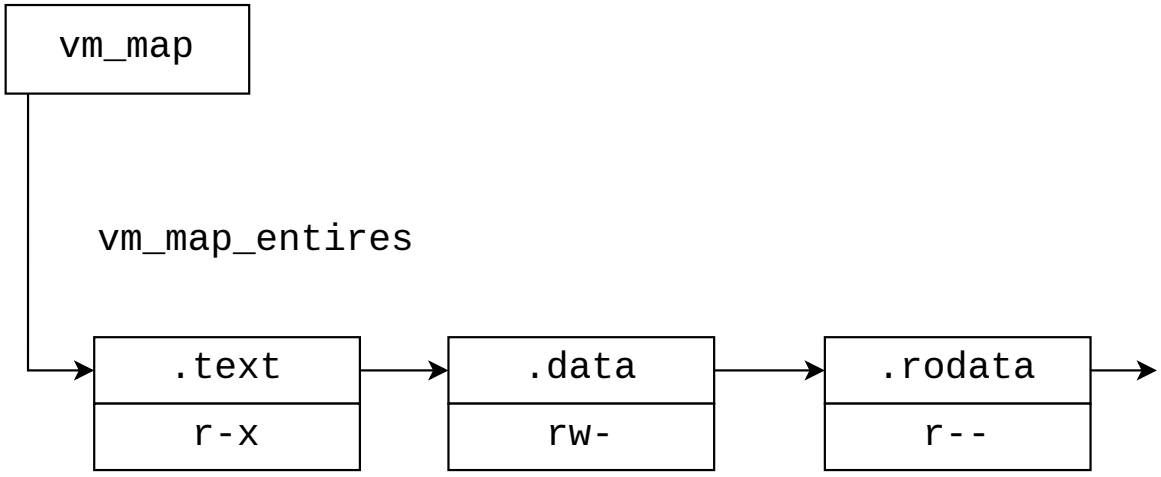
Copy-on-write (zapisy do pamięci)

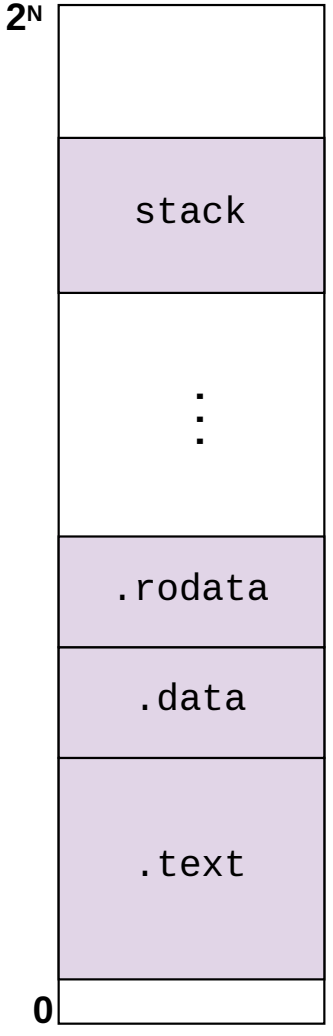


Pamięć wirtualna w systemie NetBSD (oraz Mimikerze)

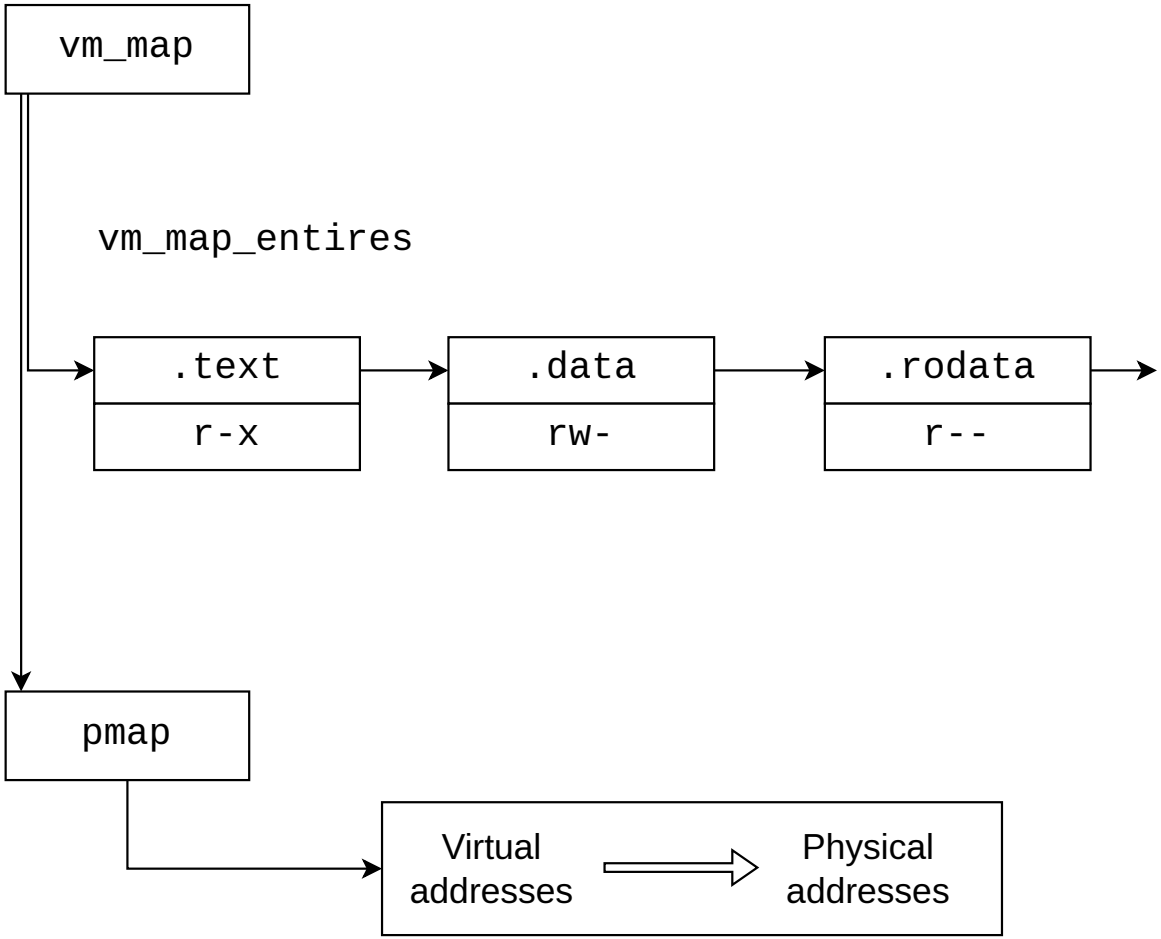
Mapa pamięci procesu

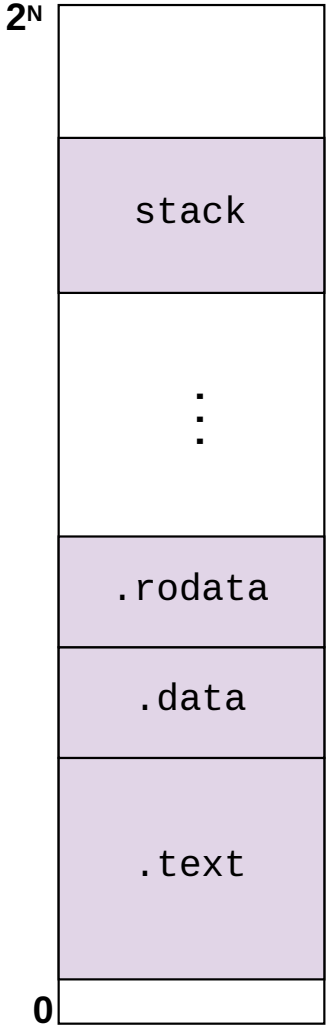
Mapa pamięci procesu





Mapa pamięci procesu





Rodzaje pamięci dostępnej w systemie

Rodzaje pamięci dostępnej w systemie

Pamięć pochodząca z zasobów systemu:

- Pliki mapowane w pamięć

Rodzaje pamięci dostępnej w systemie

Pamięć pochodząca z zasobów systemu:

- Pliki mapowane w pamięć

Pamięć anonimowa:

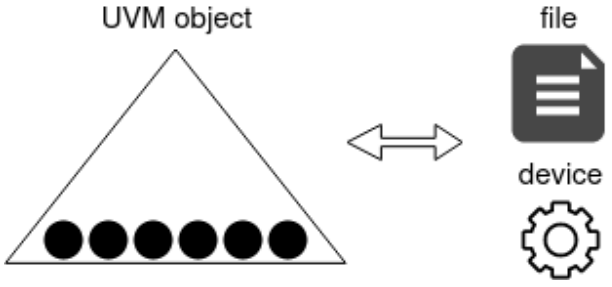
- Pamięć przydzielona procesowi
- Wypełniana prywatnymi danymi procesu
- Niszczona gdy przestaje być używana

Struktury do opisu pamięci

Struktury używane w systemie NetBSD – w systemie pamięci wirtualnej UVM.

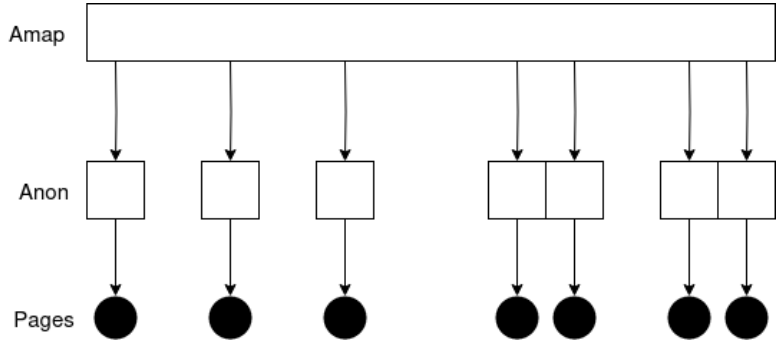
Pamięć zasobów

UVM object

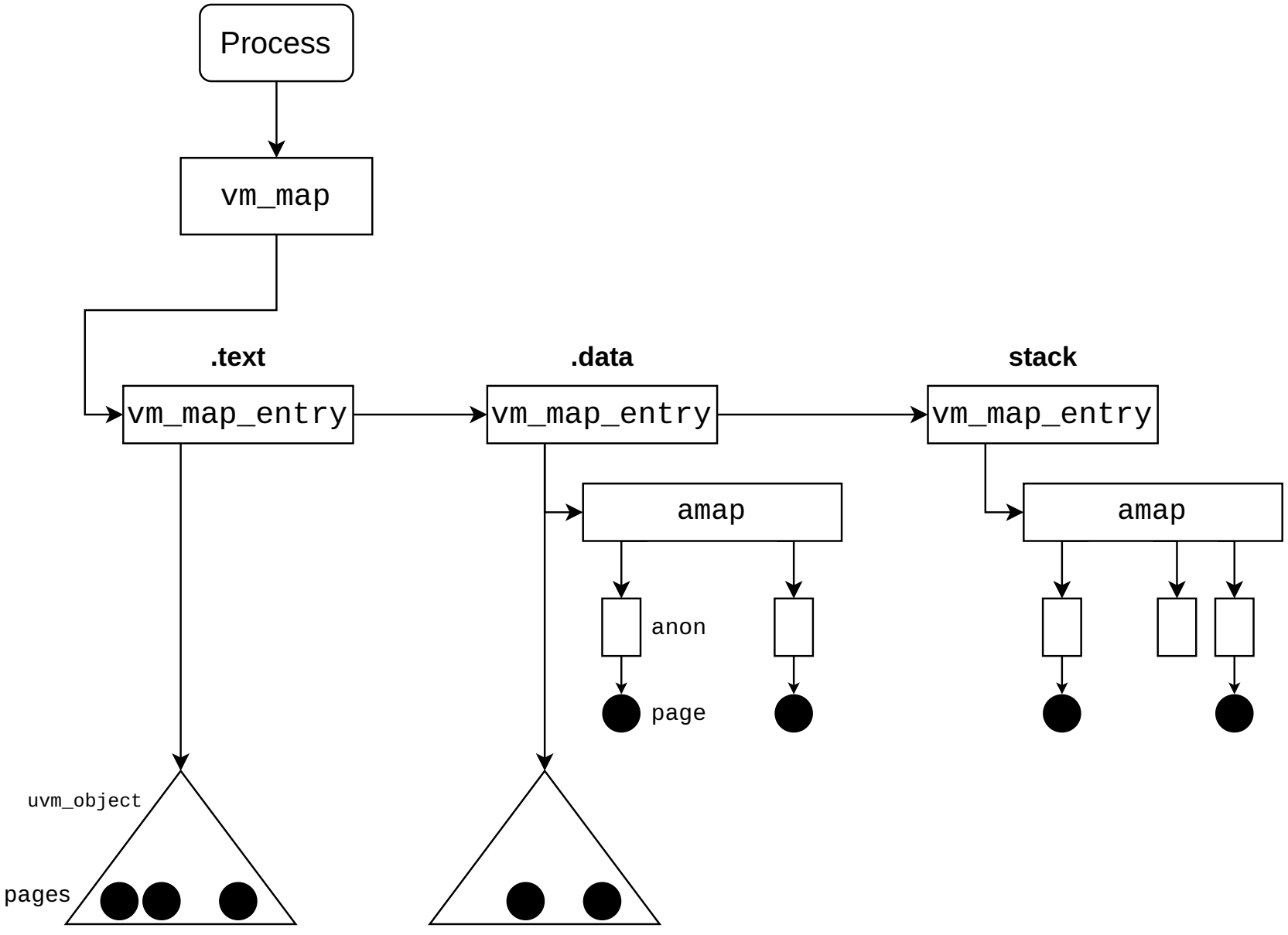


Pamięć anonimowa

Amap + Anon



Przykładowa mapa pamięci procesu



Implementacja CoW w Mimikerze

Zmiany

1. Zmiana implementacji podsystemu pamięci wirtualnej
   - BSD VM -> UVM
   - łatwiejszy do rozszerzania o nowe mechanizmy

Zmiany

1. Zmiana implementacji podsystemu pamięci wirtualnej
   - BSD VM -> UVM
   - łatwiejszy do rozszerzania o nowe mechanizmy
2. Zaimplementowanie mechanizmu copy-on-write

Zmiany

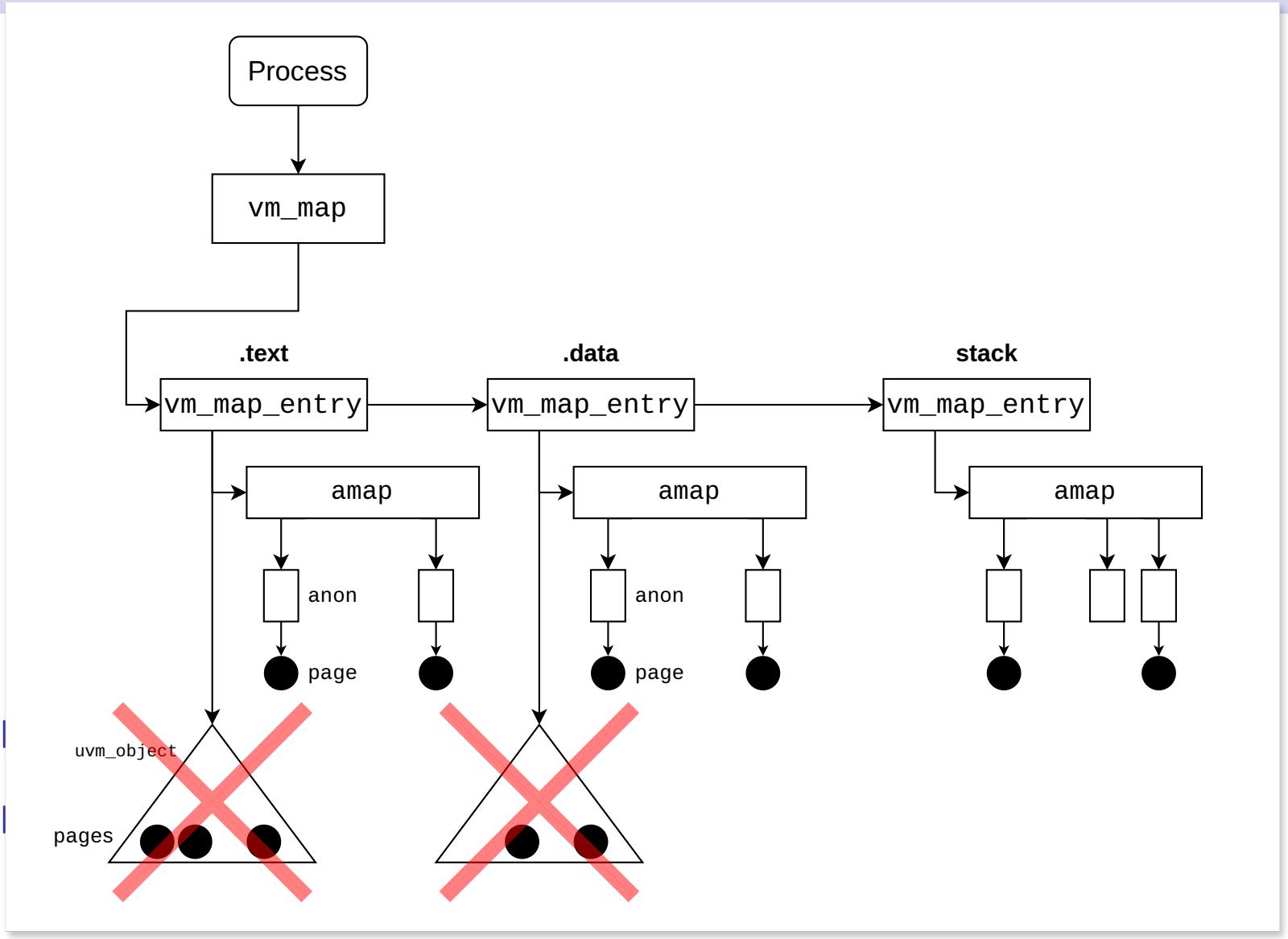
1. Zmiana implementacji podsystemu pamięci wirtualnej
   - BSD VM -> UVM
   - łatwiejszy do rozszerzania o nowe mechanizmy
2. Zaimplementowanie mechanizmu copy-on-write
3. Narzędzie do analizy wydajności jądra
   - Kernel Function Trace

Ważna obserwacja

W Mimikerze występuje tylko pamięć anonimowa.

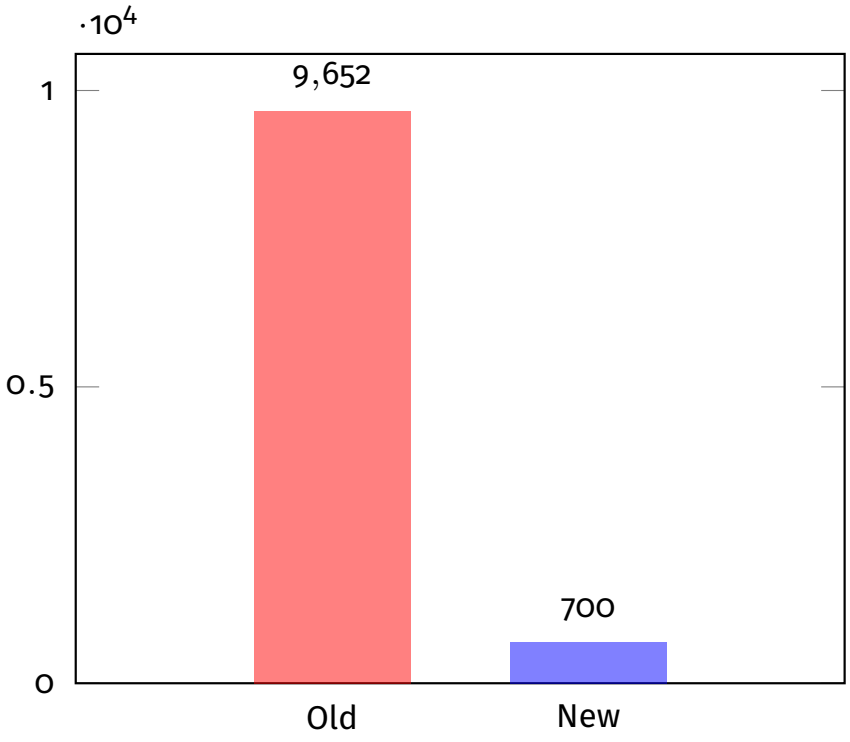
- Nie ma plików mapowanych w pamięć
- Nie funkcjonuje wymiana stron

Ważna obserwacja

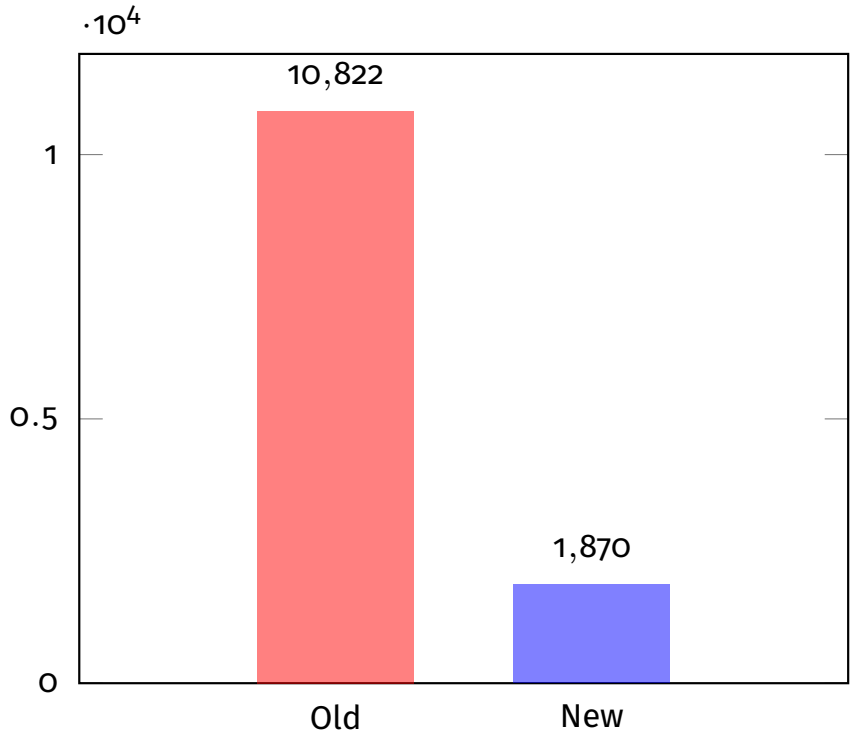


Podsumowanie analizy wydajności

Redukcja liczby kopiowanych stron

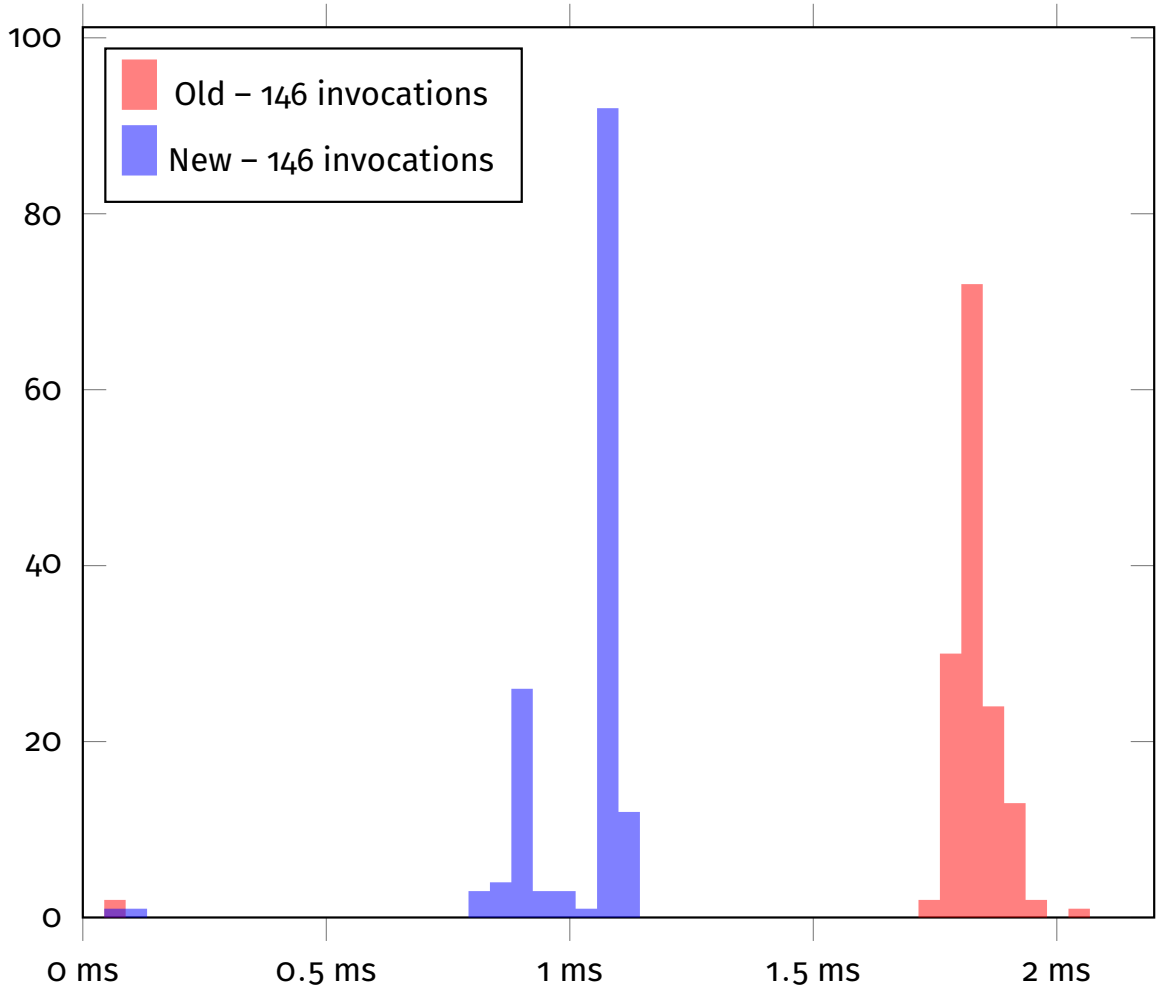


Liczba kopiowań



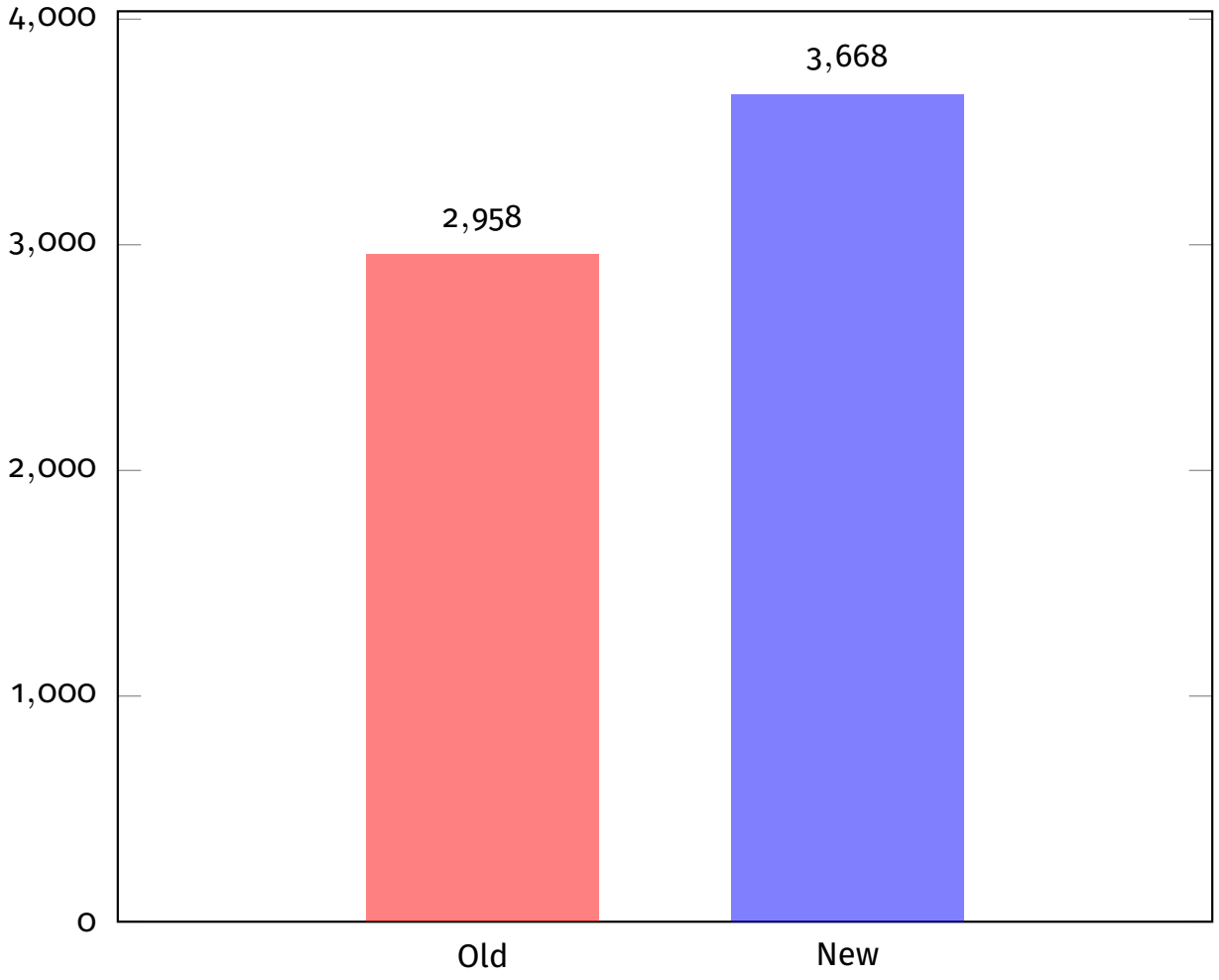
Sumaryczna liczba użytych stron

Tworzenie nowego procesu



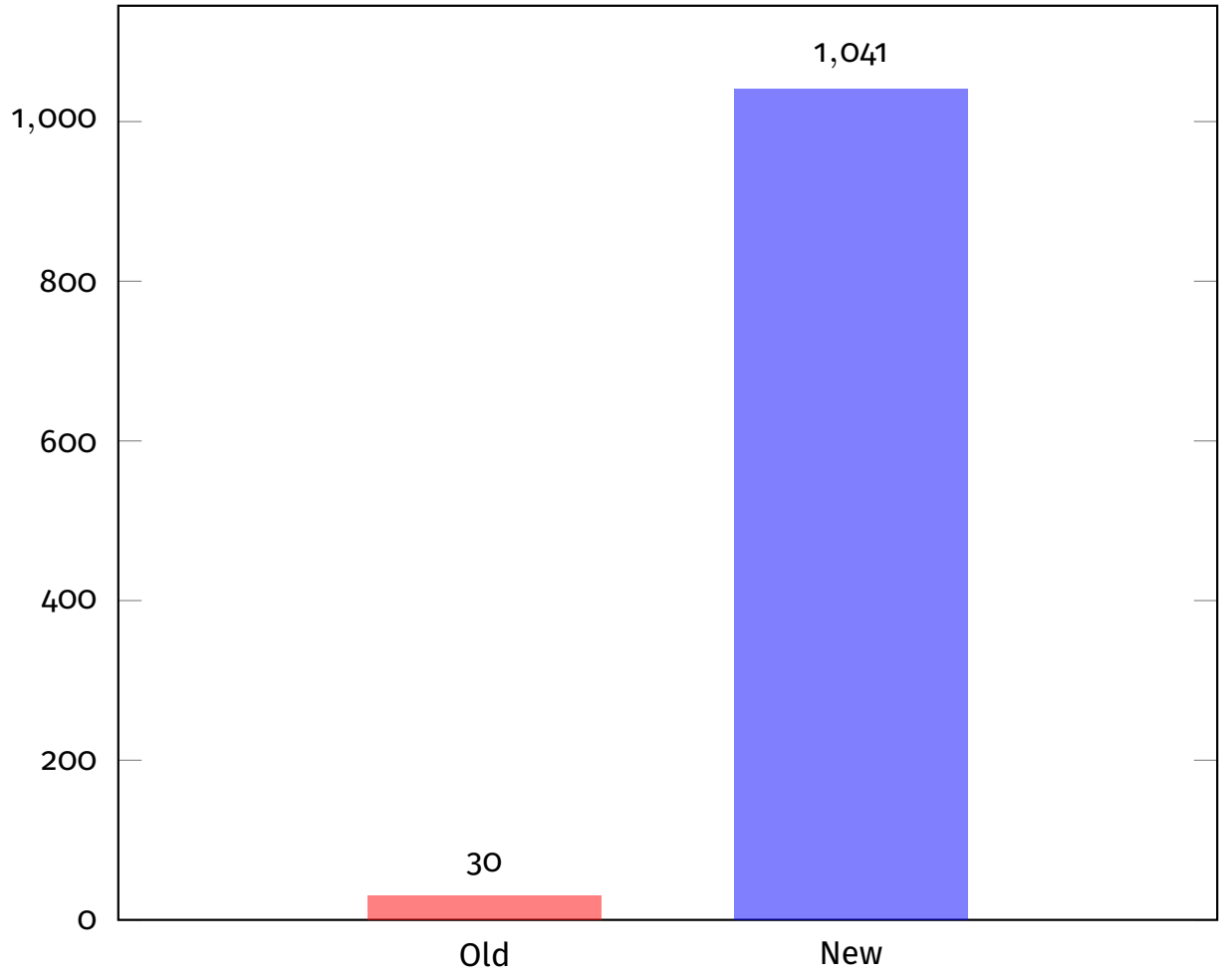
Histogram czasu wywołań dla funkcji `vm_map_clone`

Spodziewane zmiany



Ilość page faultów

Spodziewane zmiany (2)



Zmiana praw dostępu do stron

Podsumowanie

Podsumowanie wyników pracy

1. Implementacja mechanizmu copy-on-write
   - Oszczędność pamięci

Podsumowanie wyników pracy

1. Implementacja mechanizmu copy-on-write
   - Oszczędność pamięci
2. Ulepszenie implementacji systemu pamięci wirtualnej
   - Łatwiejsza do rozszerzenia o nowe funkcje
   - Łatwiejszy do analizy

Podsumowanie wyników pracy

1. Implementacja mechanizmu copy-on-write
   - Oszczędność pamięci
2. Ulepszenie implementacji systemu pamięci wirtualnej
   - Łatwiejsza do rozszerzenia o nowe funkcje
   - Łatwiejszy do analizy
3. Narzędzie do analizy wydajności (KFT)
   - Wykrywanie operacji wymagających optymalizacji

Podsumowanie wyników pracy

1. Implementacja mechanizmu copy-on-write
   - Oszczędność pamięci
2. Ulepszenie implementacji systemu pamięci wirtualnej
   - Łatwiejsza do rozszerzenia o nowe funkcje
   - Łatwiejszy do analizy
3. Narzędzie do analizy wydajności (KFT)
   - Wykrywanie operacji wymagających optymalizacji

Kolejne kroki:

- Implementacja urządzeń i plików mapowanych w pamięć
- Implementacja wymiany stron