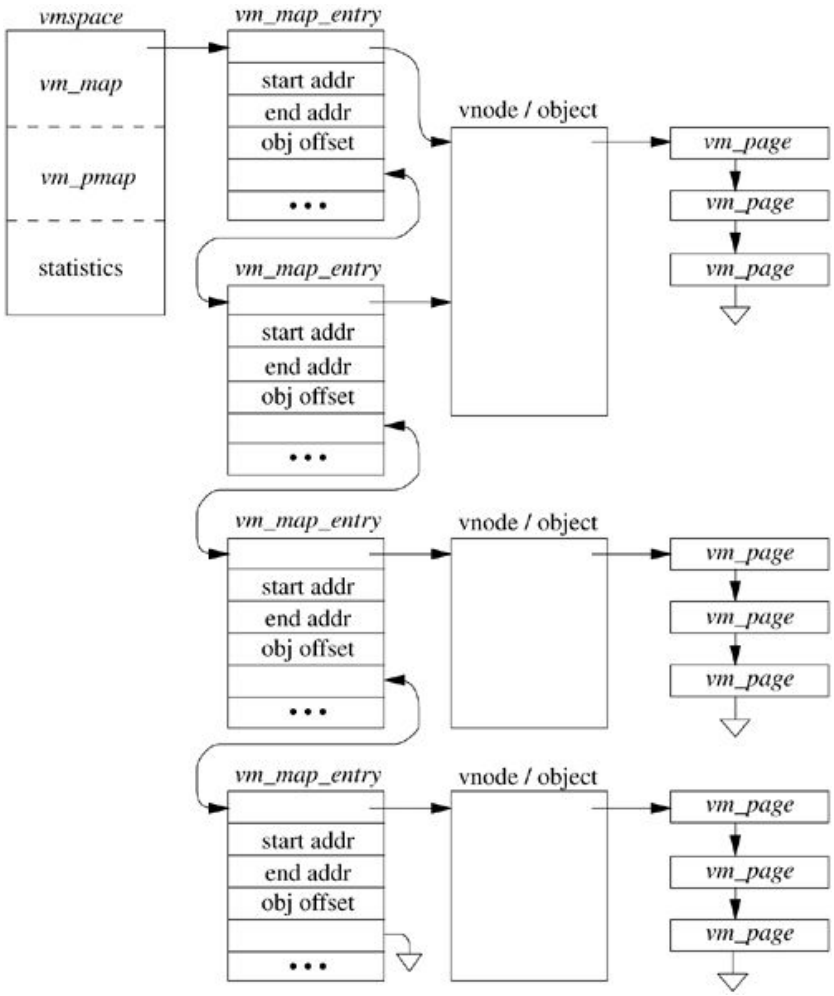
Systemy operacyjne

Ćwiczenia 6

Należy przygotować się do zajęć czytając następujące rozdziały książek:

- Stallings: 8.1 – 8.3
- Tanenbaum: 3.3 – 3.7
- Silberschatz: 9.1 – 9.9

Poniższy rysunek reprezentuje przykładową strukturę danych używaną do przechowywania informacji o przestrzeni adresowej procesu:



Struktura ta zgrubnie odpowiada temu co można znaleźć w pliku `maps` w katalogu `/proc/${pid}`. Tzn. przestrzeń adresowa to lista segmentów (`vm_map_entry`) wraz z ich uprawnieniami. Każdy segment ma przypisany obiekt (`vm_object`) wraz z odpowiadającym mu programem stronicującym (`pager`) i stronami obecnymi w pamięci głównej. Pager sprowadza strony z urządzenia, pliku, pamięci wymiany, itd. i przypisuje im ramki pamięci fizycznej. Struktury `vm_page` przechowują dane skojarzone ze stroną i wskaźnikiem na ramkę. Aby skonstruować tablicę stron (zależną od sprzętu) przechowywaną w polu `vm_pmap`, należy przejrzeć całą strukturę `vm_map`.

Więcej szczegółów można znaleźć w książce *"The Design and Implementation of the FreeBSD Operating System"*, ale zaglądanie tam nie powinno być niezbędne do zrobienia zadań z tej listy.

Zadanie 1

Wyjaśnij czym różni się **strona** od **ramki**. Znajdź informacje dotyczące **bitów pomocniczych** w **deskryptorach stron** i **deskryptorach katalogów stron** dla architektury `x86`. Które z nich dotyczą:

- sposobu używania pamięci podręcznej,
- wspomagają algorytmy zarządzania pamięcią wirtualną,
- określają uprawnienia dostępu (włączając w to tryb pracy procesora).

Opisz dokładnie strukturę **PTE** i **PDE** oraz znaczenie wszystkich bitów.

Zadanie 2

Rozważmy **procedurę obsługi braku strony**, do której przekazywane jest sterowanie w wyniku niewłaściwego odwołania do pamięci. Jakie dane musi otrzymać ta funkcja by prawidłowo wykonywać swoją pracę? Zaproponuj pseudokod procedury umożliwiającej **stronicowanie na żądanie** z uwzględnieniem uprawnień dostępu do stron. Wykorzystaj przykładową strukturę danych i następującą funkcję programu stronicującego:

- `vm_page* get_page(vm_object* object, int offset);`

Zadanie 3

Zauważ, że dzięki wyabstrahowaniu programu stronicującego w obrębie tego samego mechanizmu można zrealizować zarówno przydział pamięci, **wymianę pamięci** i **mapowanie plików** na pamięć. Czym różni się **pomniejszy błąd strony** od **głównego błędu strony**? Spróbuj naszkicować w jaki sposób będą działały programy stronicujące dla: pamięci anonimowej (prywatnej i współdzielonej), urządzeń blokowych i plików.

Zadanie 4

Wyjaśnij zasadę działania mechanizmu **kopiowania przy zapisie** w systemach ze stronicowaniem. Jakie korzyści przynosi? Do czego używa się go w praktyce? Jak rozszerzyć procedurę obsługi braku stron przy założeniu, że wprowadzono dodatkowy typ obiektu `vm_object` do śledzenia zmian (ang. *shadow object*) względem oryginalnego obiektu. Zauważ, że `vm_object` można zagnieżdżać.

Zadanie 5

Rozważmy cztery klasy polityk zarządzania pamięcią wirtualną. Wyjaśnij kiedy mają zastosowanie **polityka ładowania**, **przydziału miejsca**, **zastępowania** i **usuwania**? Poznaliśmy już stronicowanie na żądanie. Czym różni się od niego **stronicowanie wstępne** i kiedy ma sens?

Zadanie 6

Wykaż, że zasada lokalności dotyczy również pamięci wirtualnej opartej na stronicowaniu. Wyjaśnij pojęcia **zbioru roboczego** i **zestawu rezydentnego**. Jak te zbiory zmieniają się w czasie działania procesu i jak to się ma do zasady lokalności? Znajdź i opisz wykres zależności braków stron od rozmiaru ramki dla typowego programu. Czemu w systemach z bardzo małymi ramkami występuje (statystycznie) mniej braków stron niż w systemach z ramkami o średniej wielkości? Załóż, że ilość dostępnych stron rośnie odwrotnie proporcjonalnie do ich wielkości.

Zadanie 7

Rozważmy **buforowanie stron** – tj. jeśli strony zostały oznaczone do zastąpienia w wyniku stosowania konkretnej strategii (np. `FIFO`), to nie są usuwane od razu, ale trzyma się je przez jakiś czas w pamięci operacyjnej. Wymień zalety tego rozwiązania w kontekście lokalności i zarządzania zbiorem roboczym, oraz polityki usuwania stron. Jak z buforowania stron może skorzystać algorytm zarządzający **zbiorem rezydentnym zmiennego rozmiaru**, działający w obrębie całego systemu? Do czego służy **przypinanie stron** w pamięci operacyjnej?

Zadanie 8

W systemie uruchomiony jest wątek jądrowy **demona wymiany** (ang. *pageout daemon*). Jakie kryteria może on stosować do dobierania wielkości zbioru roboczego dla programu? Opisz popularny algorytm zastępowania stron **WSClock**. W jakich warunkach powstaje zjawisko **szamotania**? Co mógłby zrobić demon wymiany, aby to zjawisko wykryć i mu zapobiec?

Zadania do wykonania w domu

Rozwiązania poniższych zadań należy zapisać na kartce formatu A4, podpisać imieniem i nazwiskiem, oraz zdać na początku zajęć prowadzącemu.

Zadanie 9

Rozważmy system ze **stałym przydziałem zbioru rezydentnego**. Pamięć fizyczna składa się z 4 ramek, a pamięć wirtualna z 8 stron. Na maszynie wykonuje się jeden proces i generuje następujący ciąg dostępów do stron: `7 0 1 2 0 3 0 4 2 3 0 3 2`. Pokaż działanie następujące algorytmy zastępowania ramek: **`FIFO`**, **`LRU`**, **`CLOCK`**. Jak te algorytmy korzystają z pomocniczych bitów w deskryptorach stron? Który z nich generuje najmniej **braków stron**? Załóż, że zastąpienie ramki zachodzi dopiero w momencie zapełnienia całej pamięci fizycznej.