Warszawa, 06.01.2020

student: Patryk Jan Sozański
grupa: 2I5

SYSTEMY OPERACYJNE: LABORATORIUM NR 5
RAPORT Z WYKONANEGO ĆWICZENIA

Treść zadania

Celem ćwiczenia jest zmiana domyślnego algorytmu przydziału pamięci w systemie Minix. Należy umożliwić wybór algorytmu wyboru bloku z listy bloków wolnych między standardowym first fit a tzw. algorytmem worst fit, czyli takim, w którym wybierany jest blok pamięci z listy wolnych bloków o największym rozmiarze.

Należy zdefiniować dwie dodatkowe funkcje systemowe, identyfikowane stałymi HOLE_MAP oraz WORST_FIT.

Funkcja systemowa HOLE_MAP powinna umożliwiać zdefiniowanie własnej funkcji o sygnaturze:

int hole_map(void ∗buffer , size_t nbytes)

która ma za zadanie zwrócić w buforze buffer o rozmiarze nbytes informacje o aktualnej zawartości listy wolnych bloków utrzymywanej przez moduł zarzadzania pamięcią (MM). Struktura otrzymanej w buforze informacji powinna być następująca:

rozmiar1, adres1, rozmiar2, adres2, ..., 0

gdzie kolejne pary rozmiar-adres odpowiadają informacjom o kolejnych elementach listy wolnych bloków. Rozmiar 0 oznacza ostatni element listy. Elementy rozmiar i adres mają typ danych unsigned int (na poziomie modułu MM synonim tego typu o nazwie phys_clicks).
Funkcja hole_map ma zwracać przesłaną liczbę par rozmiar-adres. Należy zabezpieczyć się przed przepełnieniem zadanego jako argument wywołania bufora i wypełnić go tylko liczbą par mieszczących się w buforze dbając o zakończenie listy pozycją rozmiar = 0.

Funkcja systemowa WORST_FIT powinna umożliwiać wybór algorytmu wyboru elementu z listy wolnych bloków i zdefiniowanie własnej funkcji o sygnaturze:

int worst_fit (int w)

która dla w = 1 wymusza implementowany w ramach ćwiczenia algorytm przydziału worst fit, natomiast dla w = 0 uaktywnia z powrotem standardowy algorytm first fit. Wartością zwracaną powinno być zawsze 0.

Sposób rozwiązania problemu

Aby poprawnie zrealizować polecenie, należało zaimplementować algorytm WORST_FIT, zapewnić użytkownikowi systemu możliwość zmiany algorytmu oraz udostępnić dwa dodatkowe wywołania systemowe.

W pierwszej kolejności w mm.h zdefiniowałem dwie stałe symbolizujące wybrany algorytm alokacji pamięci. W tym samym pliku zadeklarowałem zmienną, która przechowuje informacje o aktualnie wybranym sposobie alokacji pamięci.

Algorytm worst fit zaimplementowałem w pliku alloc.c w funkcji alloc_mem. W tym samym pliku zdefiniowałem dwie funkcje realizujące funkcjonalności nowych wywołań systemowych. Wywołanie HOLE_MAP realizuje funkcja do_hole_map(), a drugie wywołanie systemowe WORST_FIT umożliwiające zmianę algorytmu alokacji, jest realizowane przez funkcję do_worst_fit().

Testowanie

Testy rozwiązania zrealizowałem za pomocą przykładowych programów i skryptów zaprezentowanych na stronie prowadzącego przedmiot dr inż. Tomasza Kruka. Test opierał się na trzech programach: x - symulującego program realizujący obliczenia będące de facto okrojoną wersją polecenia sleep, t - wyświetlającego liczbę i rozmiar bloków wolnych oraz w - przyjmującego jako argument wywołania 1 albo 0, włączając lub wyłączając algorytm worst fit w systemie operacyjnym. Kod źródłowy tych programów został zamieszczony na stronie prowadzącego przedmiot.

Programy testowe zostały wykorzystane w skrypcie skrypt, który pokazuje działanie mechanizmu alokacji w przypadku zastosowania obu algorytmów. Podstawowa wersja skryptu również znajduje się na stronie

prowadzącego. W celu lepszej prezentacji działania algorytmu, dokonałem kilku prostych modyfikacji. Po pierwsze oprócz wielkości dziury pamięci, prezentowany jest również adres początku dziury. Po drugie dodałem wywołanie programu t wypisującą mapę pamięci również przed pierwszą iteracją pętli uruchamiającej programy x. Dzięki temu możemy stwierdzić, czy ilość zalokowanej pamięci na zakończenie testu jest taka sama jak przed uruchomieniem pierwszego.

a) Algorytm first fit

Algorytm first fit alokuje pamięć w pierwszym wolnym kawałku pamięci, który ma wystarczający rozmiar. Dla potrzeb testów możemy założyć, że implementacja algorytmu first fit jest poprawna, ponieważ została ona zrealizowana przez twórcę systemu i wyniki testu traktować jako referencyjne. W pierwszej kolejności sprawdziłem, czy ilość wolnej pamięci przed testem i po jest taka sama. Ilość wolnej pamięci w obu momentach była równa. Co więcej, wszystkie wolne segmenty miały takie same rozmiary i ich początki znajdowały się w tych samych miejscach. Zgodnie z oczekiwaniami, liczba segmentów wolnej pamięci była stała. Co każdą iterację, z najmniejszego wolnego segmentu większego niż 9 clicków, zabierane jest dokładnie 9 clicków pamięci na rzecz uruchamianego programu x. W drugiej pętli sprawdzane są mapy pamięci w momencie, gdy programy zwalniają swoją pamięć. Pojawia się nowa dziura, która powstała, ponieważ elementy zaczynają zwalniać swoją pamięć w kolejności, w której ją zaalokowały.

b) Algorytm worst fit

Algorytm worst fit alokuje pamięć w największym znalezionym kawałku pamięci. W tym przypadku pamięć powinna być zawsze alokowana w ostatnim segmencie, który jest największy. Zgodnie z oczekiwaniami, co iterację pętli testu z największego segmentu jest pobierany kolejny kawałek na program. Między zaalokowaną pamięcią pojawiają się dodatkowe wolne segmenty wielkości 62 clicków. W algorytmie first fit ten efekt nie pojawiał się. Efekt ten jest konsekwencją sposobu uruchamiania procesów. W pierwszej kolejności jest alokowana pamięć dla kopii procesu testowego za pomocą polecenia fork, a następnie program jest podmieniany przez polecenie z grupy exec, alokując przy tym wymaganą ilość pamięci. W przypadku tego algorytmu alokacji, system operacyjny alokując pamięć dla nowego procesu, zawsze wybiera największy wolny segment i to z niego pobiera kawałek pamięci. W momencie gdy procesy zwalniają pamięć, dziury łączą się ze sobą, tworząc większe segmenty, dzięki czemu w ostatnim obiegu ilość dziur jest taka, jak była przed pierwszą pętlą tego testu. Dziury, które już były w pamięci przed testem, pozostały niezmienione, ich wielkość i pierwszy adres pozostały takie same. Pomiędzy dziurami tworzonymi w trakcie testu, odległości między początkami wynoszą 71 clicków. Każda z dziur ma 62 clicki szerokości. Wynika z tego, że pomiędzy końcem jednej dziury a początkiem drugiej jest zaalokowanych dokładnie 9 clicków dla procesu x. Jest to dokładnie taka sama ilość alokowanej pamięci jak w przypadku referencyjnego algorytmu first fit. W drugiej pętli, pamięć procesów x jest zwalniana, co objawia się powiększaniem dziury pozostałej po pierwszej iteracji pierwszej pętli.

Wynik działania skryptu testującego

```
x: Stack+malloc area changed from 8000 to 8000 bytes.
-[ std ]--------------------------------
[7]     5:112   17:123  38:157  46:320  62:435  28:559  128563:1357
[7]     5:112   6:134   38:157  46:320  62:435  28:559  128563:1357
[7]     5:112   6:134   29:166  46:320  62:435  28:559  128563:1357
[7]     5:112   6:134   20:175  46:320  62:435  28:559  128563:1357
[7]     5:112   6:134   11:184  46:320  62:435  28:559  128563:1357
[7]     5:112   6:134   2:193   46:320  62:435  28:559  128563:1357
[7]     5:112   6:134   2:193   37:329  62:435  28:559  128563:1357
[7]     5:112   6:134   2:193   28:338  62:435  28:559  128563:1357
[7]     5:112   6:134   2:193   19:347  62:435  28:559  128563:1357
[7]     5:112   6:134   2:193   10:356  62:435  28:559  128563:1357
[7]     5:112   6:134   2:193   1:365   62:435  28:559  128563:1357
[7]     5:112   15:125  2:193   1:365   62:435  28:559  128563:1357
[8]     5:112   15:125  9:157   2:193   1:365   62:435  28:559  128563:1357
[8]     5:112   15:125  18:157  2:193   1:365   62:435  28:559  128563:1357
[8]     5:112   15:125  27:157  2:193   1:365   62:435  28:559  128563:1357
[7]     5:112   15:125  38:157  1:365   62:435  28:559  128563:1357
[8]     5:112   15:125  38:157  9:320   1:365   62:435  28:559  128563:1357
[8]     5:112   15:125  38:157  18:320  1:365   62:435  28:559  128563:1357
[8]     5:112   15:125  38:157  27:320  1:365   62:435  28:559  128563:1357
[8]     5:112   15:125  38:157  36:320  1:365   62:435  28:559  128563:1357
[7]     5:112   17:123  38:157  46:320  62:435  28:559  128563:1357
-[ worst ]------------------------------
[8]     5:112   17:123  38:157  46:320  62:435  28:559  62:1222 128501:1419
[9]     5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 128428:1492
[10]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 128357:1563
[11]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 62:1437 128286:1634
[12]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 62:1437 62:1508 128215:1705
[13]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 62:1437 62:1508 62:1579 128144:1776
[14]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 62:1437 62:1508 62:1579 62:1650 128073:1847
[15]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 62:1437 62:1508 62:1579 62:1650 62:1721 128002:1918
[16]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 62:1437 62:1508 62:1579 62:1650 62:1721 62:1792 127931:1989
[17]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 62:1437 62:1508 62:1579 62:1650 62:1721 62:1792 62:1863 127860:2060
[18]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 62:1295 62:1366 62:1437 62:1508 62:1579 62:1650 62:1721 62:1792 62:1863 62:1934 127789:2131
[18]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 71:1286 62:1366 62:1437 62:1508 62:1579 62:1650 62:1721 62:1792 62:1863 62:1934 127789:2131
[17]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 142:1286        62:1437 62:1508 62:1579 62:1650 62:1721 62:1792 62:1863 62:1934 127789:2131
[16]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 213:1286        62:1508 62:1579 62:1650 62:1721 62:1792 62:1863 62:1934 127789:2131
[15]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 284:1286        62:1579 62:1650 62:1721 62:1792 62:1863 62:1934 127789:2131
[14]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 355:1286        62:1650 62:1721 62:1792 62:1863 62:1934 127789:2131
[13]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 426:1286        62:1721 62:1792 62:1863 62:1934 127789:2131
[12]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 497:1286        62:1792 62:1863 62:1934 127789:2131
[11]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 568:1286        62:1863 62:1934 127789:2131
[10]    5:112   17:123  38:157  46:320  62:435  28:559  62:1222 639:1286        62:1934 127789:2131
[8]     5:112   17:123  38:157  46:320  62:435  28:559  62:1222 128501:1419
-[ std ]--------------------------------
```