Aleksandra Sikora

Uniwersalny symulator algorytmów planowania procesów

4 stycznia 2017

Spis treści

1 Opis zadania **3**

1.1 Wprowadzenie 3

1.2 Algorytm FCFS 4

1.3 Algorytm SJF 5

1.4 Planowanie priorytetowe 6

1.5 Planowanie rotacyjne 7

2 Urachamianie i testowanie **8**

1 Opis zadania

1.1 Wprowadzenie

Proces (nazywany też czasem zadaniem) to wykonujący się program. Wykonanie instrukcji jednego procesu musi być sekwencyjne, co znaczy, że instrukcje są wykonywane w określonej kolejności. Każdy proces ma własny licznik instrukcji, który wskazuje następną instrukcję do wykonania oraz własny obszar przydzielonej mu pamięci operacyjnej. W każdej chwili proces jest w jakimś stanie. Ten stan zmienia się w miarę postępu wykonania procesu.

Stany procesu:

- nowy,
- aktywny,
- czekający,
- gotowy,
- zakończony.

Planowanie procesów polega na wskazywaniu procesu któremu ma być w danej chwili przydzielony procesor. W systemie w każdej chwili może być aktywnych co najwyżej tyle procesów ile jest procesorów. Wszystkie pozostałe procesy muszą czekać na przydział procesora. Celem planowania jest maksymalizacja czasu wykorzystania procesora przy wieloprogramowości. Większość procesów działa w cyklu złożonym na przemian z fazy procesora (okresu, w którym proces wykonuje obliczenia) i fazy wejścia-wyjścia (okresu, w którym proces czeka na ukończenie zleconej operacji wejścia-wyjścia).

Strategie planowania można oceniać z różnych punktów widzenia. Oto możliwe kryteria planowania:

- wykorzystanie procesora,
- przepustowość,
- czas oczekiwania,
- czas cyklu przetwarzania,
- czas odpowiedzi.

Planowanie przydziału procesora jest to określenie kolejności przydziału jego mocy obliczeniowej procesom z kolejki procesów gotowych do działania. Poniżej zostaną omówione wybrane algorytmy planowania przydziału procesora.

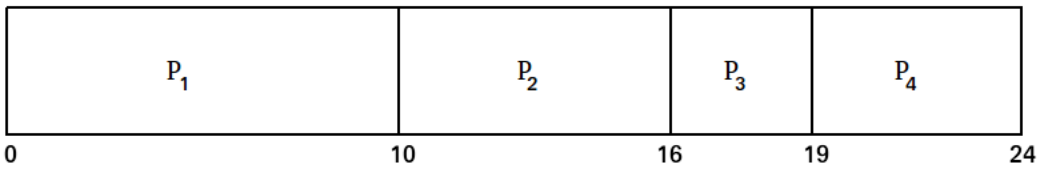
1.2 Algorytm FCFS

Algorytm FCFS (ang. *first-come, first-served* – FCFS), czyli „pierwszy zgłoszony – pierwszy obsłużony ", to strategia szeregowania bez wywłaszczania. Procesy są wykonywane od początku do końca w takiej kolejności, w jakiej pojawiły się w kolejce procesów gotowych do wykonania.

Rozważymy przyjście w chwili 0 czterech procesów, których długości faz procesora wynoszą:

Proces	Czas trwania fazy
P_1	10
P_2	6
P_3	3
P_4	5

Nadejście procesów w kolejności P_1, P_2, P_3, P_4, obsłużonych w porządku FCFS wyglądać będzie następująco na diagramie Gantta:



Czasy oczekiwania wynoszą odpowiednio 0,10,16 oraz 19 ms dla procesów P_1, P_2, P_3, P_4. Zatem średni czas oczekiwania wynosi $\frac{0+10+16+19}{4} = 11,25$ ms.

Widać więc, że FCFS nie jest najlepszą strategią szeregowania, ponieważ średni czas oczekiwania jest duży. Może się również zdarzyć, że procesy długo oczekują na zwolnienie procesora przez jeden wielki proces. Taką sytuację nazywamy **efektem konwoju** (ang. *convoy effect*). Efekt ten powoduje mniejsze wykorzystanie procesora i urządzeń.

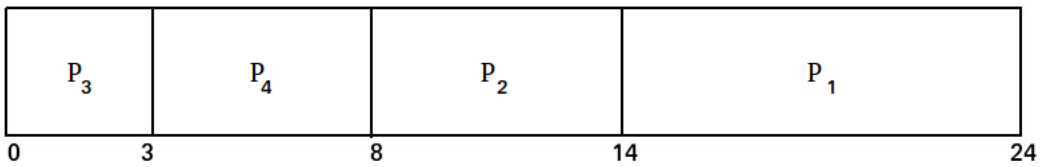
1.3 Algorytm SJF

Algorytm SJF (ang. *shortest-job-first* – SJF), czyli „najkrótsze zadanie najpierw", opiera swe działanie na powiązaniu procesu z długością jego najbliższej fazy. Procesor zostaje przydzielony procesowi, który ma najkrótszą następną fazę procesora. W przypadku, gdy dwa procesy mają równe fazy procesora, stosuje się algorytm FCFS.

Porównajmy działanie tego algorytmu do algorytmu FCFS. Rozważmy jako przykład cztery procesy, ktrórych długości faz procesora wynoszą:

Proces	Czas trwania fazy
P_1	10
P_2	6
P_3	3
P_4	5

Diagram Gantta dla tych procesów wygląda następująco:



Czasy oczekiwania wynoszą odpowiednio 0,3,8 oraz 14 ms dla procesów P_1, P_2, P_3, P_4. Zatem średni czas oczekiwania wynosi $\frac{0+3+8+14}{4} = 6,25$ ms.

Algorytm SJF daje minimalny średni czas dla danego zbioru procesów. Mówimy, że algorytm SJF jest optymalny.

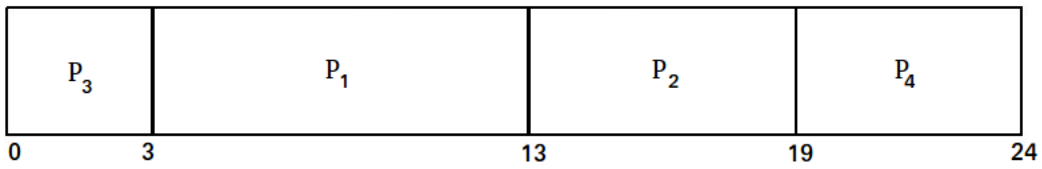
1.4 Planowanie priorytetowe

Algorytm planowania priorytetowego (ang. *priority scheduling*) jest ogólnym przypadkiem algorytm SJF. Zamiast długości następnej fazy z algorytmu SJF, procesowi przypisuje się priorytet. Mniejsze wartości oznaczają wyższy priorytet. Procesy o wyższym priorytecie mają pierwszeństwo w dostępie do procesora przed procesami o priorytecie niższym. Procesom o równych priorytetach przydziela się procesor zgodnie z algorytmem FCFS. Priorytety definiuje się wewnętrznie lub zewnętrznie. Wewnętrzna definicja priorytetu opiera się na własnościach procesu. Do określenia priorytetów zewnętrznych używa się kryteriów spoza wnętrza systemu, np.: kwota opłat za użytkowanie komputera, znaczenie procesu dla użytkownika, itp.

Rozważmy następujący zbiór procesów przybyłych w chwili 0:

Proces	Czas trwania fazy
P_1	10
P_2	6
P_3	3
P_4	5

Przy zastosowaniu planowania priorytetowego otrzymujemy uporządkowanie takie jak na poniższym diagramie Gantta:



Czasy oczekiwania wynoszą odpowiednio 0,3,13 oraz 19 ms dla procesów P_1, P_2, P_3, P_4. Zatem średni czas oczekiwania wynosi $\frac{0+3+13+19}{4} = 8,75$ ms.

Problem związany z planowaniem priorytetowym to **nieskończone blokowanie** (ang. *indefinite blocking*), nazywane **głodzeniem** (ang. *starvation*). Problem wiąże się z odkładaniem możliwości obsługi przez procesor tych procesów, które mają niski priorytet. Rozwiązaniem jest **postarzanie** (ang. *aging*) procesów o niskich priorytetach, które długo oczekują na obsługę procesora. Odbywa się ono co pewien czas. W ten sposób z procesu o niskim priorytecie, tworzy się proces o wysokim priorytecie, który na pewno zostanie obsłużony w pierwszej kolejności.

1.5 Planowanie rotacyjne

Algorytm planowania rotacyjnego (ang. *round-robin* – RR) jest oparty na algorytmie FCFS, jednak różni się od FCFS tym, że możliwe jest wywłaszczanie. Algorytm ten jest chętnie stosowany w systemach z podziałem czasu. Z każdym procesem powiązany jest kwant czasu, najczęściej 10 do 100 ms. Procesom znajdującym się w kolejce (traktowanej jako cykliczna) procesów gotowych do wykonania, przydzielane są odpowiednie odcinki czasu, nie dłuższe niż jeden kwant czasu.

Rozważmy następujący przykład:

Proces	Czas trwania fazy
P_1	10
P_2	6
P_3	3
P_4	5

Dla kwantu czasu wynoszącego 4 ms otrzymamy następujący diagram Gantta:

Czasy oczekiwania wynoszą odpowiednio 14,15,8 oraz 17 ms dla procesów P_1, P_2, P_3, P_4. Średni czas oczekiwania wynosi: $\frac{14+15+17+8}{4} = 13,5$ ms.

Wadą tej strategi jest dość duży średni czas oczekiwania i częste przełączanie kontekstu. Na podanym wyżej przykładzie potrzebnych było siedem przełączeń. Natomiast zaletą planowania rotacyjnego jest prostota tej strategii oraz krotki czas reakcji, szczególnie ważny przy interakcyjnej pracy wielu użytkowników.

2 Urachamianie i testowanie

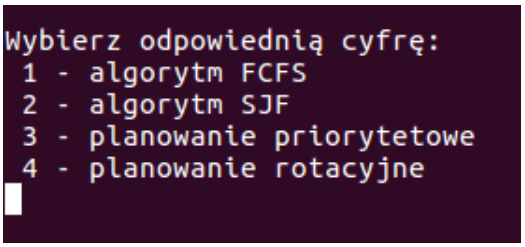
Do uruchomienia programu wymagany jest system Linux. W celu skompilowania programu należy użyć polecenia:

gcc -o c main.c

Następnie uruchamiany program poleceniem:

./c

Po uruchomieniu programu mamy wybór spośród czterech omówionych wcześniej algorytmów planowania przydziału procesora:



Następnie musimy podać liczbę procesów, dla których chcemy sprawdzić działanie danego algorytmu oraz czasy trwania faz każdego z nich. W przypadku algorytmu priorytetowego dodatkowo trzeba nadać każemu z procesów priorytet, a przy planowaniu rotacyjnym podać kwant czasu. Poniżej znajduje się przykład działania planowania priorytetowego: