Systemy operacyjne

wykład 8 - zarządzanie procesami cz. 2

dr Marcin Ziółkowski

Instytut Matematyki i Informatyki
Akademia im. Jana Długosza w Częstochowie

5 maja 2016 r.

STRATEGIE PLANOWANIA

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). Planując przydział procesora, powinniśmy to przewidzieć tak, aby nie okazało się, że w wyniku strategii planowania np. wszystkie procesy czekają na wejście-wyjście i żaden nie jest gotowy do korzystania z procesora.
Planista wybiera spośród procesów gotowych ten, który ma być wykonywany, i przydziela mu procesor. Jeśli decyzje planisty są podejmowane jedynie w następujących chwilach:

- przejście procesu aktywnego do stanu oczekiwania
- zakończenie procesu

to planowanie nazywamy **niewywłaszczeniowym** (nie ma sytuacji, w której planista swoją decyzją wstrzymuje proces aktywny).

Decyzje planisty mogą być podejmowane także w takich chwilach:

- przejście procesu ze stanu oczekiwania do stanu gotowości
- przejście procesu aktywnego do stanu gotowości

Wówczas mówimy o planowaniu **wywłaszczeniowym** (są sytuacje, w których planista swoją decyzją wstrzymuje proces aktywny).

EKSPEDYTOR

Ekspedytor to proces egzekwujący wyroki planisty krótkoterminowego. Przekazuje wybranemu procesowi władzę nad procesorem, co wymaga przełączenia kontekstu, przejścia w tryb użytkownika i wykonania skoku do instrukcji, którą teraz należy wykonać (np. do tej instrukcji, przed którą ów proces poprzednio wywłaszczono). Niestety czas pracy ekspedytora nie jest zerowy i jest to czas poświecony na systemową biurokrację. W tym czasie żaden proces użytkownika nie wykonuje swoich instrukcji, więc prace nie posuwają się do przodu. Czas potrzebny na zatrzymanie procesu i uruchomienie innego nazywamy **opóźnieniem ekspedycyjnym**. Im jest ono mniejsze, tym lepiej. Wynika stąd, że planista nie może swych decyzji podejmować zbyt często, ponieważ wówczas narzuty systemowe będą ogromne ("biurokracja udusi gospodarkę").

KRYTERIA JAKOŚCI PLANOWANIA

Strategie planowania można oceniać z różnych punktów widzenia. Przyjęcie konkretnego kryterium może całkowicie zmienić naszą ocenę danej strategii. Oto możliwe kryteria:

- **wykorzystanie procesora** – Oczywiste kryterium. Dbaj o to, żeby procesor był jak najlepiej wykorzystywany.
- **przepustowość** – Przepustowość to liczba faz procesora wykonanych na jednostkę czasu. Należy ją maksymalizować.
- **czas oczekiwania** – Czas oczekiwania to czas spędzony przez proces w kolejce gotowych. Najlepiej, żeby był jak najkrótszy.
- **czas obrotu** – Czas obrotu to czas wykonywania jednej fazy procesora (pobytu procesu w stanie gotowy i aktywny). Powinien być jak najkrótszy. Czas obrotu jest sumą czasu oczekiwania (który zależy od szeregowania) oraz czasu wykonywania obliczeń w danej fazie procesora (który nie zależy od szeregowania).
- **czas reakcji** – Czas reakcji to czas od zajścia zdarzenia, na które proces ma zareagować, do powzięcia pierwszej reakcji (nie zakończenia cyklu obliczeń). Należy go również minimalizować. Kryterium to dotyczy systemów interakcyjnych z podziałem czasu.

FCFS (First-Come, First-Served), czyli "pierwszy przyszedł pierwszy obsłużony", to strategia szeregowania bez wywłaszczania. Procesy są wykonywane od początku do końca w takiej kolejności, w jakiej pojawiły się w kolejce gotowych. Przypuśćmy, że w chwili $t = 0$ trzy procesy P1, P2 i P3 pojawiły się w kolejce procesów gotowych, w tej właśnie kolejności, a ich wykonanie trwało odpowiednio 6, 3 i 2 jednostek czasu. Proces P2 rozpoczął więc działanie w chwili $t = 6$, a proces P3 w chwili $t = 9$ (po zakończeniu procesów P1 i P2). Proces P1 czekał 0 jednostek czasu, P2 czekał 6 jednostek, a P3 czekał 9 jednostek czasu. Średni czas oczekiwania wynosił zatem $(0 + 6 + 9) / 3 = 5$. Natomiast czas obrotu procesów wynosi odpowiednio: 6, 9 i 11 jednostek, co daje średni czas obrotu $(6 + 9 + 11) / 3 = 8{,}67$.

Gdyby planista ustawił te procesy w innej kolejności (np. P3, P2 a na końcu P1), to okresy oczekiwania wynosiłyby: dla P1 - 5 jednostek czasu, dla P2 - 2 jednostki czasu, a dla P3 - 0 jednostek czasu. Średni czas oczekiwania wynosiłby zatem $(5 + 2 + 0) / 3 = 2{,}33$. Można więc było osiągnąć ponad dwukrotnie krótszy czas średni czas oczekiwania! Analogicznie, czasy obrotu wynosiłyby wtedy: dla P1 - 11 jednostek czasu , dla P2 - 5 jednostek, a dla P3 - 2 jednostki, co daje średni czas obrotu $(11 + 5 + 2) / 3 = 6$. Widać więc, że FCFS nie jest najlepszą strategią. Jej zaletą jest za to prostota, co wiąże się z niskimi narzutami systemowi (planista nie ma zbyt wiele do roboty).

SJF

SJF (Shortest-Job-First), czyli "najpierw najkrótsze zadanie" to strategia, w której decyzje planisty zależą od przewidywanych długości następnych faz procesora gotowych procesów. Jest to strategia bez wywłaszczania. Gdy jakiś proces zwolni procesor, planista wybiera z kolejki gotowych ten proces, którego przewidywana długość następnej fazy procesora jest najkrótsza. Załóżmy, że w chwilach czasu $0, 2, 4, 5$ w kolejce gotowych procesów pojawiły się procesy P1, P2, P3 oraz P4. Przewidywane długości faz procesora dla tych procesów wynoszą odpowiednio: 8, 5, 1 oraz 2 jednostki czasu.

Wówczas w chwili $t = 0$ jest tylko jeden proces gotowy (P1) i on właśnie jest wykonywany pierwszy. W chwili $t = 8$ (gdy P1 zwolni procesor) już wszystkie pozostałe procesy są gotowe, więc planista wybiera proces o najkrótszej przewidywanej następnej fazie procesora, czyli P3. Po jego zakończeniu planista wznawia kolejno P4 i P2. Proces P1 czeka 0 jednostek czasu, P2 czeka 9 jednostek, P3 czeka 4 jednostki i P4 też 4 jednostki. Średni czas oczekiwania wynosi zatem $(0 + 9 + 4 + 4) / 4 = 4{,}25$. Natomiast czasy obrotu wynoszą: dla P1 - 8 jednostek, dla P2 - 14, dla P3 - 5, a dla P4 - 6 jednostek czasu. Tak więc średni czas obrotu wynosi: $(8 + 14 + 5 + 6) / 4 = 8{,}25$.

SRTF

SRTF (ang. Shortest-Remaining-Time-First), czyli "najpierw zadanie o najkrótszym pozostałym czasie wykonania" to wersja strategii SJF z wywłaszczaniem. W strategii tej decyzja planisty jest oczywiście podejmowana w chwili zakończenia fazy procesora procesu aktywnego, ale również w chwili, gdy którykolwiek proces zmienia stan na gotowy. W takiej chwili wybiera się proces, którego pozostały przewidywany czas fazy procesora jest najkrótszy. Jeśli w trakcie wykonywania procesu pojawiły się inny gotowy proces, którego oczekiwany czas fazy procesora jest krótszy niż pozostały oczekiwany czas działania procesu aktywnego, to proces aktywny jest wywłaszczany a w jego miejsce procesor otrzymuje nowo przybyły proces. Rozważmy ten sam przykład, co poprzednio, ale dla strategii SRTF.

Gdy w chwili $t = 2$ przybywa proces P2, pozostały czas działania P1 wynosi 6 jednostek czasu, a oczekiwany czas dla P2 to 5 jednostek. Proces P1 jest więc **wywłaszczany**, a procesem aktywnym staje się P2. Podobnie dzieje się, gdy pojawia się proces P3, który powoduje wywłaszczenie P2. W chwili, gdy kończy się P3, przybywa P4 z oczekiwanym czasem fazy procesora równym 2 jednostki, więc to właśnie on a nie P1 albo P2 staje się aktywny. Po jego zakończeniu wznawiane są kolejno P2 i P1. Okresy oczekiwania wynoszą: dla P1 – $0 + 8 = 8$ jednostek , dla P2 – $0 + 3 = 3$ jednostki, a dla P3 i P4 – 0 jednostek. Średni czas oczekiwania wynosi zatem $(8 + 3 + 0 + 0) / 4 = 2{,}75$. Natomiast czasy obrotu wynoszą: dla P1 – 16, dla P2 – 8, dla P3 – 1 i dla P4 – 2 jednostki. Tak więc, średni czas obrotu wynosi $(16 + 8 + 1 + 2) / 4 = 6{,}75$. Osiągnęliśmy więc znacznie lepszy wynik niż w wypadku strategii SJF.

SZACOWANIE FAZY PROCESORA

Ktoś mógłby powiedzieć, że to tylko przykład, jednak można udowodnić, że SRTF jest strategią optymalną pod względem minimalizacji średniego czasu oczekiwania oraz minimalizacji średniego czasu obrotu. Istnieje jednak mały szkopuł. Dokładne określenie długości fazy procesora wymaga przewidywania przyszłości. Nie da się więc zaimplementować tej strategii. Zobaczymy jednak jak można, w przybliżeniu, oszacować długość fazy procesora. Nie wiemy ile będzie trwała kolejna faza procesora, ale wiemy ile trwały poprzednie. Okazuje się, że kolejne fazy procesora są zwykle podobnej długości. Przy tym albo proces ma charakter bardziej interaktywny i jego fazy procesora są krótkie, albo ma charakter obliczeniowy i jego fazy procesora są długie. Jedną ze stosowanych metod szacowania następnej fazy procesora na podstawie długości poprzednich faz jest **uśrednianie wykładnicze**.

SZACOWANIE FAZY PROCESORA

Stosując tą metodę, długość kolejnej fazy procesora szacujemy jako średnią ważoną długości poprzednich faz procesora, przy czym wagi tworzą malejący ciąg geometryczny. Przyjmijmy następujące oznaczenia:

- $o(n)$ to oszacowanie długości n-tej fazy procesora
- $r(n)$ to rzeczywista długość n-tej fazy procesora
- a to waga długości ostatniej fazy w średniej ważonej

Należy ustalić wartość współczynnika a z przedziału $(0, 1]$. Ten współczynnik decyduje o tym, jak ważna jest długość ostatniej fazy procesora w stosunku do starszych faz. Gdy $a = 1$, kolejne oszacowanie jest po prostu równe długości poprzedniej fazy. Wagi w naszej średniej ważonej tworzą ciąg:

$$a, a(1-a), a(1-a)^2, a(1-a)^3 \ldots.$$

W ten sposób otrzymujemy następujący wzór rekurencyjny:

$$o(n+1) = ar(n) + (1-a)o(n).$$

Rozwiązanie jest następujące:

$$o(n) = a\sum_{i=1}^{n-1} r(n-i)(1-a)^{i-1} + o(1)(1-a)^{n-1}.$$

Jak widać współczynniki wagi czasów coraz starszych faz procesora maleją wykładniczo. Stąd nazwa "uśrednienie wykładnicze"

PLANOWANIE ROTACYJNE

RR (Round-Robin), czyli planowanie rotacyjne, to strategia, w której każdy proces po kolei otrzymuje kwant czasu procesora do wykorzystania. Zwykle jest to około 20-100 milisekund. Żaden proces nie czeka więc dłużej na procesor niż długość kwantu razy liczba procesów. Planowanie rotacyjne powoduje znacznie więcej przełączeń kontekstu jest więc dość kosztowną strategią. Kwant czasu musi być co najmniej o rząd wielkości większy niż czas przełączenia kontekstu. W przeciwnym wypadku prace administracyjne (biurokracja systemowa) zduszą rzeczywistą pracę wykonywaną przez system dla użytkowników.

PLANOWANIE ROTACYJNE

Popatrzmy na zastosowanie planowania rotacyjnego w przykładzie, który wykorzystaliśmy przy okazji omówienia strategii SJF. Przyjmijmy, że kwant czasu wynosi 2. Oto jak wykonywałyby się te procesy (procesy gotowe są obsługiwane w takiej kolejności, w jakiej ustawiają się w kolejce procesów gotowych, a więc jest to kolejka FIFO). Okresy oczekiwania wynoszą: dla P1 – $2 + 5 + 1 = 8$, dla P2 – $3 + 4 = 7$, dla P3 – 2, a dla P4 – 4. Średni czas oczekiwania wynosi zatem $(8 + 7 + 2 + 4) / 4 = 5{,}25$. Osiągnęliśmy więc gorszy wynik niż w wypadku pozostałych strategii. Jak widać z punktu widzenia średniego czasu oczekiwania jest to bardzo zła strategia. Tym, co sprawia, że często korzysta się z planowania rotacyjnego, jest mały czas reakcji, szczególnie ważny przy interakcyjnej pracy wielu użytkowników, oraz prostota tej strategii.

Przy strategiach tego typu każdy proces ma statycznie (albo dynamicznie) przydzielany priorytet. Procesor jest przydzielany procesowi, który ma największy priorytet. Tu również można mówić o strategiach niewywłaszczeniowych (gdy proces dobrowolnie opuszcza procesor) i wywłaszczeniowych (gdy proces jest pozbawiany procesora natychmiast po pojawieniu się procesu o wyższym priorytecie). Na strategie SJF i SRTF można spojrzeć jak na szczególny przypadek strategii priorytetowych. Priorytetem jest w ich wypadku oszacowanie długości czasu obliczeń do końca fazy procesora, traktowane jako wartość ujemna (im mniejsze oszacowanie, tym większy priorytet).

Problemem przy stosowaniu strategii priorytetowych jest **zagłodzenie**, tzn. sytuacja, w której proces może czekać w nieskończoność na swoją kolej i nigdy się nie doczekać. Stanie się tak wówczas, gdy ciągle będą pojawiać się nowe procesy o priorytecie wyższym niż ów czekający proces. Będzie on im musiał cały czas ustępować im pierwszeństwa. Rozwiązaniem tego problemu może być, na przykład, automatyczne podwyższanie priorytetu procesu w miarę jego starzenia (upływu czasu oczekiwania). Wówczas żaden proces nie czekałby w nieskończoność, bo miałby wówczas nieskończony priorytet, co jest niemożliwe.

WIELOPOZIOMOWE KOLEJKI

Dotychczas wszystkie procesy traktowaliśmy tak samo. Nie jest to uzasadnione. Wskazaliśmy strategie dobre z punktu widzenia pracy wsadowej (SJF i SRTF) oraz dobre z punktu widzenia pracy interakcyjnej (RR). Jeśli odróżnimy w systemie procesy wsadowe od interakcyjnych, będziemy mogli zastosować do każdej z tych grup inną strategię planowania, odpowiednią dla danej grupy procesów. Użytkownik może wskazać procesy tła (wsadowe) i czoła (interakcyjne). Procesy tła mogą być obsługiwane strategią SJF, ponieważ w przypadku pracy wsadowej zależy nam na minimalizacji średniego czasu oczekiwania, a procesy czoła mogą być obsługiwane strategią RR, ponieważ w przypadku pracy interakcyjnej zależy nam na minimalizacji czasu reakcji. Pozostaje jeszcze rozstrzygnąć wzajemny stosunek obu tych grup procesów. Oczywiście procesy czoła powinny być uprzywilejowane (minimalizujemy czas reakcji w ich wypadku!). W jakim jednak stopniu? Można przyjąć, że zawsze wykonujemy procesy czoła przed wsadowymi w nadziei, że zawsze pojawiają się momenty bezczynności użytkowników.

KOLEJKI ZE SPRZĘŻENIEM ZWROTNYM

Jeśli jednak obawiamy się zagłodzenia procesów tła, można przyjąć np. podział czasu między kolejki procesów tła i czoła, np. 80% dla czoła i 20% dla tła. Co zrobić, jeśli nie wiemy z góry, które procesy są interaktywne, a które nastawione na obliczenia, albo charakter procesu zmienia się w miarę jego działania. Możemy wówczas użyć kilku kolejek dopuszczając, aby procesy były między nimi przenoszone. Owe przenoszenie nazywamy sprzężeniem zwrotnym. Definicja strategii planowania z wieloma kolejkami i sprzężeniem zwrotnym musi obejmować:

1. liczbę i uszeregowanie kolejek
2. strategię szeregowania dla każdej kolejki
3. sposób decydowania o przeniesieniu procesu do innej kolejki
4. sposób decydowania o kolejce, w której należy umieścić nowy proces

Wyobraźmy sobie na przykład strategię z trzema kolejkami: Q0, Q1 i Q2, kierującą się następującymi zasadami:

KOLEJKI ZE SPRZĘŻENIEM ZWROTNYM

1. Kolejka Q0 ma wyższy priorytet niż pozostałe (tzn. jeśli w Q0 jest jakiś proces, to procesy z pozostałych kolejek nie są aktywne
2. Kolejka Q1 ma wyższy priorytet niż Q2
3. Kolejka Q0 jest obsługiwana metodą RR z kwantem 20 milisekund
4. Kolejka Q1 jest obsługiwana metodą RR z kwantem 50 milisekund
5. Kolejka Q2 jest obsługiwana metodą FCFS
6. Nowy proces trafia do Q0
7. Jeśli proces z kolejki Q0 nie zakończy działania przed upływem swojego kwantu czasu (20 milisekund), to jest wywłaszczany i przenoszony do kolejki Q1
8. Jeśli proces z kolejki Q1 nie zakończy działania przed upływem swojego kwantu czasu (50 milisekund), jest wywłaszczany i przenoszony do kolejki Q2. Jeżeli natomiast jego faza procesora jest krótsza niż 20 milisekund, to jest przenoszony do kolejki Q0
9. Jeśli proces z kolejki Q2 ma fazę procesora krótszą niż 50 milisekund, to jest przenoszony do kolejki Q1

KOLEJKI ZE SPRZĘŻENIEM ZWROTNYM

W ten sposób procesy, które często żądają wejścia-wyjścia, czyli procesy interakcyjne, są uprzywilejowane. To słuszne podejście, bo takie procesy i tak rzadko korzystają z procesora, a ta strategia pozwala ograniczyć czas ich przestoju do oczekiwania na wejście-wyjście. Dzięki temu takie procesy szybko opuszczą system, nie "przeszkadzając" przy tym specjalnie pozostałym (one rzadko zajmują procesor). Zwróćmy uwagę, że w ta strategia sama selekcjonuje procesy tła i czoła, uniezależniając tę decyzję od użytkownika (który może się mylić albo oszukiwać). Ponadto, jeżeli charakter procesu zmienia się w miarę upływu czasu, jest on automatycznie przenoszony do właściwej kolejki.