AUTOMATYKA • 2009 • Tom 13 • Zeszyt 2

Piotr Szymczyk*, Magdalena Szymczyk*

Metody harmonogramowania procesów w systemach operacyjnych czasu rzeczywistego

1. Wprowadzenie

Systemem operacyjnym nazywamy program lub układ programów, który pośredniczy pomiędzy użytkownikiem komputera a sprzętem. Do podstawowych jego zadań zaliczamy nadzór nad działaniem innych programów, koordynację dostępu do sprzętu przez programy użytkowe, dostarczanie środków do optymalnego użycia zasobów, czyli czasu procesora, obszarów pamięci operacyjnej, pamięci masowej oraz urządzeń wejścia/wyjścia [3].

Procesem nazywamy wykonujący się program wraz z jego środowiskiem obliczeniowym. Każdy proces, aby wykonać swoje zadanie, wymaga przydziału zasobów, takich jak: czas procesora, pamięć, pliki i urządzenia wejścia/wyjścia. Do najważniejszych funkcji systemu w odniesieniu do procesów, zaliczamy:

- załadowanie i wykonanie,
- zakończenie i zaniechanie,
- utworzenie i zakończenie procesu,
- pobranie i określenie atrybutów procesu,
- oczekiwanie na zdarzenie i jego sygnalizacja,
- przydział i zwolnienie pamięci.

Wyróżniamy trzy podstawowe stany procesów. Proces uruchomiony (*running*), to proces wykonujący się w danej chwili na procesorze. Proces gotowy (*ready*), jest to proces gotowy do wykonania, ale wstrzymany w oczekiwaniu na przydział czasu procesora. Proces wstrzymany (*blocked*), nie może kontynuować pracy do momentu wystąpienia pewnego zewnętrznego zdarzenia.

Przydziałem procesora poszczególnych procesom zajmuje się część jądra systemu, nazywana planistą (*scheduler*). System operacyjny posiada tablicę procesów, w której zawarte są szczegółowe informacje na temat określonego procesu, takie jak stan procesu, priorytet,

* Katedra Automatyki, Akademia Górniczo-Hutnicza w Krakowie

ID procesu i jego grupa, informacje o czasie zużytym przez proces itp. Element tablicy procesów nazywamy deskryptorem procesu lub blokiem kontrolnym procesu (PCB – *process control block*).

Główną cechą systemu operacyjnego czasu rzeczywistego jest istnienie wymagań, związanych z czasem wykonywanych przez niego operacji. System ten reaguje w sposób przewidywalny na nieprzewidywalne zdarzenia zewnętrzne. Poprawność działania systemu jest uzależniona nie tylko od poprawności uzyskanego wyniku, ale również od czasu jego uzyskania. Problem szeregowania procesów w systemach tego typu jest więc bardzo ważnym zagadnieniem. Należy określić, któremu z procesów zostanie przydzielony procesor, oraz na jak długi czas tak, aby wszystkie wykonane procesy spełniały zdefiniowane dla nich ograniczenia czasowe.

Nie wszystkie elementy systemu operacyjnego czasu rzeczywistego podlegają szeregowaniu. Wątki, procesy i zadania mogą być szeregowane, natomiast elementy takie jak obsługa przerwań, komendy systemu operacyjnego, usługi sieciowe oraz proces zarządzający szeregowaniem zadań, nie podlegają szeregowaniu.

2. Algorytmy szeregowania procesów

Rozpatrując problem szeregowania, posługujemy się dwoma podstawowymi pojęciami: zadania oraz zasoby. Zadanie polega na wykonaniu ciągu operacji, z których każda wymaga zaangażowania konkretnych zasobów.

Cechami charakterystycznymi dla zadań są: wymagany termin zakończenia, przerywalność operacji, podzielność, sposób wykonania. W przypadku zasobów musimy wziąć pod uwagę takie cechy jak: dostępność, ilość, koszt, podzielność, przywłaszczalność. Zasoby dzielimy na odnawialne i nieodnawialne. W przypadku szeregowania w procesach operacyjnych, używanymi zasobami są procesory, które należą do zasobów odnawialnych [11].

Model matematyczny, konstruowany na potrzeby rozwiązania problemu szeregowania procesów w systemach operacyjnych, powinien posiadać następujące założenia:

- dany jest zbiór zadań, z czego każde zadanie składa się z ciągu operacji,
- zadania mają zostać wykonane przy użyciu zbioru maszyn, różnego typu,
- operacje w obrębie zadania, muszą być wykonywane w ściśle określonej kolejności.

W modelu tym możemy wybrać następujące zmienne decyzyjne:

- termin rozpoczęcia wykonywania operacji,
- sposób wykonania operacji.

Przyjmujemy przy tym założenia, że wykonanie określonej operacji nie może być przerwane, oraz że każda maszyna może wykonywać co najwyżej jedno zadanie w danej chwili czasowej.

Rozwiązaniem tak postawionego problemu jest para wektorów, określających czas rozpoczęcia oraz sposób wykonania operacji. Algorytm szeregowania odpowiada więc za uporządkowanie kolejności dostępu zadań do procesora, lub inaczej, rozdziela dostępny czas procesora i dostęp do innych zasobów pomiędzy konkurujące zadania [1, 6, 12].

2.1. Cele i typy szeregowania zadań

Poprawne szeregowanie procesów, cechuje się przede wszystkim sprawiedliwością w przydziale czasu procesora. Kolejność i sposób wykonywania zadań powinny być zgodne z przyjętymi założeniami. Cechą charakterystyczną dobrego szeregowania jest również zrównoważenie, które umożliwia zbliżoną zajętość wszystkich części systemu.

Ponadto w przypadku systemów czasu rzeczywistego, szeregowanie powinno umożliwiać spełnianie ograniczeń czasowych, oraz maksymalną przewidywalność. W systemach interaktywnych bardzo waży jest czas odpowiedzi systemu oraz proporcjonalność, pozwalająca na spełnianie oczekiwań użytkownika.

Główne cele szeregowania to:

- uzyskanie zadowalającego czasu odpowiedzi,
- uzyskanie dobrej przepustowości wykonywania procesów,
- efektywne wykorzystanie procesora.

Główne typy szeregowania procesów to:

- szeregowanie bez wywłaszczania (*nonpreemptive scheduling*),
- szeregowanie z wywłaszczaniem (*preemptive scheduling*).

W zależności od wykorzystywanego środowiska (systemy przetwarzania wsadowego, systemy interaktywne, systemy czasu rzeczywistego), odpowiednio modyfikujemy szeregowanie, poprzez nałożenie konkretnych wymagań.

Wywłaszczanie jest techniką, w której planista może zatrzymać aktualnie wykonujący się proces lub wątek, aby umożliwić działanie innemu. Dzięki temu, zawieszenie jednego procesu nie może spowodować zawieszenia się systemu. Poza tym możliwe jest określenie czasu, w jakim dany proces może korzystać z procesora. Istnieją systemy, w których nie występuje wywłaszczanie, system taki musi posiadać informacje, w jakiej chwili może przejść do kolejnych zadań – w przeciwnym razie może dojść do niepoprawnego jego działania. Istnieją zadania, które nie mogą być wywłaszczane, na przykład przerwania sprzętowe.

Szeregowanie bez wywłaszczania może być użyte w sytuacji, gdy wykonujący się proces przeszedł do stanu uśpienia (w przypadku pojawienia się zgłoszenia od urządzeń wejścia/wyjścia) lub gdy proces zakończył swoją pracę. Szeregowanie z wywłaszczaniem występuje w sytuacjach, gdy wykonujący się proces przeszedł ze stanu aktywnego lub ze stanu oczekiwania do stanu gotowego.

Wywłaszczanie jest realizowane poprzez wzrost priorytetu jednego z procesów, będącego w stanie gotowy, powyżej priorytetu procesu wykonywanego. Argumentami przytoczonej funkcji priorytetu są parametry procesu oraz stan systemu operacyjnego. Zależy ona od:

- wymagań odnośnie do wielkości przestrzeni adresowej pamięci,
- czasu oczekiwania (czasu, w którym proces pozostawał w stanie gotowości),
- czasu obsługi (czasu, w którym proces był wykonywany przez procesor),
- priorytetu zewnętrznego (uzależnionego od klasy użytkownika),
- obciążenia systemu (liczby procesów przebywających w systemie).

Kolejkowanie procesów o tym samym priorytecie odbywa się na podstawie różnych reguł. Mogą więc one być wykonywane losowo, w przypadku gdy liczba procesów o jednakowym priorytecie jest niewielka. Przydział procesora kolejnym procesom może być cykliczny, można również obsługiwać procesy w kolejności FIFO (*First In First Out*), czyli w kolejności przyjmowania procesów do systemu [11].

2.2. Przegląd algorytmów szeregowania

Procesy pojawiające się w systemie trafiają do kolejki zadań. Te procesy, które są gotowe do działania, znajdują się w pamięci głównej, na liście procesów gotowych. Przełączanie procesora do innego procesu wymaga zapamiętania zarówno stanu poprzedniego procesu, jak i załadowania stanu nowego procesu (przełączanie kontekstu). W czasie wykonywania tej czynności system nie wykonuje żadnej pracy [4, 5, 10].

Podczas planowania strategii szeregowania należy pamiętać o tym, że procesy wykonują się w określonych cyklach. Wykonujący się proces przechodzi naprzemiennie od fazy procesora, do fazy wejścia/wyjścia. W ostatniej fazie działania, wysyła do systemu żądanie zakończenia.

Istnieją różne algorytmy planowania przydziału procesora, ich wyboru powinniśmy dokonywać przede wszystkim na podstawie klasy spodziewanych procesów.

Istnieje kilka strategii wyboru procesów do wykonania. Należą do nich między innymi:

- Karuzelowa – każdy z procesów jest kolejno wykonywany, aż upłynie pewien kwant czasu (np. 100 ms).
- Odwrotność reszty kwantu – im mniej części kwantu czasu zużył proces, tym wyższa jest jego pozycja w kolejce.
- Strategia karuzelowa ze sprzężeniem zwrotnym – każdy nowo przyjęty proces otrzymuje tyle czasu, ile wszystkie inne procesy w systemie; następnie stosujemy kolejkowanie karuzelowe.
- Priorytetowa – wybierany jest proces o najwyższym priorytecie.
- Ograniczona karuzelowa – procesy są wybierane metodą karuzelową przez określoną liczbę razy. Po jej wykorzystaniu, mogą być uruchamiane tylko wtedy, gdy w systemie nie ma innych zleceń.

– Równowaga systemu – pierwszeństwo mają procesy, które wykonują dużą liczbę operacji wejścia/wyjścia.
– Lepsze traktowanie procesów interakcyjnych – proces wywołany bezpośrednią komunikacją z użytkownikiem, wykonuje się jako pierwszy.

Ponadto system może automatycznie przydzielać najwyższe priorytety zadaniom krótkim. Możliwe jest również równoważenie obciążenia tak, aby jednocześnie wykonywać jedno zlecenie z dużym zapotrzebowaniem na urządzenia wejścia/wyjścia, a drugie z dużym obciążeniem procesora centralnego.

Szeregowanie bez wywłaszczania pozwala na osiągnięcie szybkiej przeciętnej reakcji dla małych zadań. Stosujemy reguły priorytetów, opierające się całkowicie na oszacowaniu czasu obsługi podanym przez użytkownika.

Do algorytmów tego typu należą:

– FCFS (*First-Come, First-Served*) – pierwszy zgłoszony, pierwszy obsłużony;
– SJF (*Shortest Job First*) – procesy o najkrótszej kolejnej fazie wykorzystania procesora są wykonywane jako pierwsze (możliwy w wersji wywłaszczeniowej i niewywłaszczeniowej);
– Planowanie priorytetowe.

Planowanie z wywłaszczaniem redukuje problem istnienia procesów o bardzo długim czasie wykonania – daje gwarancję reakcji dla małych zadań. Jednak wywłaszczanie komplikuje budowę i działanie systemu operacyjnego – system musi jednocześnie śledzić wiele prac na różnym poziomie zaawansowania. Zamiana prac w pamięci wewnętrznej, jest głównym źródłem kosztów dla systemów z wywłaszczaniem.

Do algorytmów wywłaszczających zaliczamy:

– planowanie rotacyjne (*round robin*),
– planowanie wielokolejkowe.

Planowanie priorytetowe oraz SJF również posiadają wersje z wywłaszczaniem. W przypadku algorytmu SJF jego wersję wywłaszczeniową nazywa się również SRTF (*Shortest-Remaining-Time-First*).

2.3. Ocena i porównanie algorytmów szeregowania

Ocena przydatności algorytmów planowania może opierać się na kilku kryteriach:

– Wykorzystanie procesora – dążymy do tego, aby procesor był stale zajęty. W rzeczywistym systemie powinien być on wykorzystywany w przedziale 40–90%.
– Przepustowość – definiowana jako liczba procesów skończonych w jednostce czasu.
– Czas cyklu przetwarzania – czas pomiędzy nadejściem procesu do systemu a jego zakończeniem.

– Czas oczekiwania – czas, który proces spędza w kolejce procesów gotowych.
– Czas odpowiedzi – czas pomiędzy wysłaniem żądania a pojawieniem się pierwszej odpowiedzi (nie obejmuje czasu potrzebnego na wyprowadzenie odpowiedzi).

Porównując przedstawione parametry dla różnych algorytmów, możemy przyjąć wartości średnie, ekstremalne lub wariancję wartości (dążymy do jej minimalizacji).

Oceny algorytmów szeregowania możemy dokonać analitycznie, poprzez symulację bądź przez implementację i testowanie.

Ocena analityczna polega na stworzeniu odpowiedniego modelu matematycznego dla danego algorytmu i obciążenia roboczego systemu. Pozwala na uzyskanie przybliżonej oceny algorytmów w wyidealizowanym systemie. Symulacja jest podobną metodą – programuje się model zachowania programu – sztuczny lub stworzony na podstawie wykonania programu w rzeczywistym systemie. Dokładność metody jest uzależniona od stopnia złożoności symulacji [11].

2.4. Przegląd systemów operacyjnych pod kątem szeregowania procesów

SYSTEMY OPERACYJNE CZASU RZECZYWISTEGO

Strategia planowania w systemach operacyjnych czasu rzeczywistego jest uzależniona od rodzaju systemu. Dla systemów rygorystycznych (wypełnianie krytycznych zadań w zagwarantowanym czasie) procesy są dostarczane wraz z określeniem wymagań czasowych. Na tej podstawie planista albo akceptuje proces i zapewnia wykonanie go na czas, albo odrzuca zadanie jako niewykonalne. Łagodne systemy czasu rzeczywistego zapewniają wyższy priorytet dla procesów o decydującym znaczeniu. W tym przypadku planowanie musi być priorytetowe, a procesy czasu rzeczywistego muszą mieć najwyższy priorytet. Ich priorytety nie mogą maleć z upływem czasu.

Istnieje kilka metod szeregowania w systemach operacyjnych tego typu.

Metoda zegarowa polega na podejmowaniu decyzji o wykonaniu zadania w konkretnie ustalonych chwilach czasu. Decyzje są podejmowane periodycznie i sterowane sprzętowym zegarem zewnętrznym. Jest to najczęściej wykorzystywana metoda w systemach czasu rzeczywistego.

Metoda priorytetowa obejmuje dużą grupę algorytmów szeregowania. W metodzie tej dąży się do zapewnienia ciągłej pracy procesora i minimalizacji czasu oczekiwania zadań na wykonanie. Decyzje o przydzieleniu zadania są podejmowane, gdy zachodzą określone zdarzenia, na przykład uwolnienie czy zakończenie zadania.

Kolejną grupę stanowią algorytmy EDF (*Earliest Deadline First*) i LST (*Least Slack Time*) – optymalne algorytmy dla zadań wywłaszczalnych, które nie walczą między sobą o inne zasoby [2, 7, 8].

System RTLinux

RTLinux to rozszerzenie systemu operacyjnego Linux, które pozwala na wykonanie zadań o wymaganiach systemu czasu rzeczywistego. W systemie tym jądro czasu rzeczywistego współistnieje z jądrem tradycyjnego systemu Linux. Zadania RTLinuxa znajdują się we wspólnej przestrzeni adresowej, lecz mają maksymalny poziom uprzywilejowania.

Planista systemu implementuje algorytm oparty o stałe priorytety zadań. Wybierane jest zadanie gotowe do wykonania i o najwyższym priorytecie (RMS – *Priority Based Rate Monotonic Scheduling Algorithm*). Jeśli w systemie pojawi się kilka zadań o tym samym, najwyższym priorytecie i są one gotowe do wykonania, wybierane jest to, które zostało odnalezione jako pierwsze. Zadanie samo oddaje procesor, lub zostaje wywłaszczone przez inne o wyższym priorytecie. Planista jest uruchamiany tylko wtedy, gdy istnieje taka potrzeba (nie uruchamia się co pewien określony przedział czasu), co zmniejsza narzuty związane z jego działaniem.

Priorytety są nadawane na podstawie czasu działania danego zadania (im krótszy, tym wyższy priorytet). Wadą tego algorytmu jest fakt, że nie zawsze gwarantuje on wykonanie wszystkich zadań na czas.

Kolejnym algorytmem wykorzystywanym w omawianym systemie jest algorytm EDF. W tym przypadku zadanie o bliższym nieprzekraczalnym terminie wykonania otrzymuje wyższy priorytet. Wadą tej metody są duże narzuty związane z obliczaniem priorytetów. Zaletą jest maksymalne wykorzystanie procesora.

Obydwa spośród wymienionych algorytmów nie gwarantują wypełnienia w czasie zadań nieokresowych, które pojawiają się w systemie w dowolnym czasie. Do obsługi tego typu zadań przeznaczone są algorytmy Slot Shifting i Stack Sterling [11].

System VxWorks

VxWorks to system operacyjny oparty na wielozadaniowości, komunikacji między zadaniami oraz na procedurach obsługi przerwań.

W przypadku tego systemu zastosowany został algorytm szeregowania priorytetowego z wywłaszczaniem. System posiada 256 poziomów priorytetów (0 – priorytet najwyższy, 255 – najniższy). Priorytety są przypisywane na etapie tworzenia zadania, jednak ich wartość może się zmieniać w trakcie wykonania. Dzięki temu możliwe jest dostosowanie priorytetów zadań do aktualnych potrzeb.

System VxWorks posiada możliwość rozszerzenia algorytmu szeregowania do systemu rotacyjnego. Umożliwia to współdzielenie czasu procesora przez zadania o tym samym priorytecie. W przypadku działania tego algorytmu, jeśli pojawi się zadanie o takim samym priorytecie, aktywne zadanie wykonuje się tylko przez określony przedział czasu, a następnie zwalnia procesor dla kolejnego zadania. Jeśli w trakcie wykonywania się procesu pojawi się inny o wyższym priorytecie, następuje wywłaszczenie na rzecz tego procesu. Po jego zakończeniu, wywłaszczony proces wykorzystuje pozostały mu należny czas procesora [13].

Windows CE (Windows Embedded)

Nazwą Windows Embedded określa się dwa systemy operacyjne: Windows XPE oraz Windows CE. Windows CE jest rygorystycznym systemem czasu rzeczywistego, przeznaczonym do systemów wbudowanych.

Do szeregowania zadań w tym systemie wykorzystywany jest algorytm wywłaszczający oparty o priorytety i podział czasu. Każdy wątek jest wykonywany przez określony kwant czasu (100 ms), tak więc mamy do czynienia z algorytmem karuzelowym. Priorytety są przydzielane statycznie [9].

3. Podsumowanie

W artykule przedstawiono zagadnienia związane z problemem szeregowania zadań w systemach operacyjnych, w tym w systemach operacyjnych czasu rzeczywistego. Dobór odpowiedniego algorytmu do specyfiki projektowanego systemu jest zasadniczą kwestią decydująca o pracy całego systemu. W artykule zawarto również wskazówki pozwalające na porównanie różnych algorytmów i wybranie najbardziej odpowiedniego do danych potrzeb.

Literatura

[1] Cormen T.H., Leiserson C.E., Rnald L. Rivest T. L., *Wprowadzenie do algorytmów*. WNT, Warszawa 2001.
[2] Cottet F., Delacroix J., Kaiser C., Mammeri Z., *Scheduling in real-time systems.* Willey 2002.
[3] galaxy.uci.agh.edu.pl/~kca/R12_System_operacyjny.doc (Jakubowska M., *Systemy operacyjne – wprowadzenie*).
[4] http://www.cs.put.poznan.pl/sop/ (Instytut Informatyki Politechniki Poznańskiej, Zakład Systemów Informatycznych).
[5] http://www.prz.rzeszow.pl/we/katedry/zsc/projekty/2000/0/index.html (Politechnika Rzeszowska, „Organizacja procesów obliczeniowych").
[6] http://th-www.if.uj.edu.pl/~placzek/dydaktyka/SO/wyklady/wyklad04k.pdf (Płaczek W., *Planowanie przydziału procesora*).
[7] http://www.kik.pcz.pl/so-add/RTOS/index.htm (Rojek M., *Systemy czasu rzeczywistego*).
[8] http://users.uj.edu.pl/~ufbodek/RTS/RTS.htm (Bodek K., *Systemy czasu rzeczywistego – wykład specjalistyczny*).
[9] http://students.mimuw.edu.pl/SO/Projekt05-06/temat1-g2/szeregowanieXP.html (Moszczyński M., *Szeregowanie procesów w systemie Windows XP*).
http://www.iit.pwr.wroc.pl/~kolacz/PWR_Unix/UNIX_4_b.pdf (Kołaczek G., *Środowisko UNIX – zarządzanie procesami*).
Olender J., Kępa E., *Algorytmy szeregowania procesów w systemach operacyjnych*. Projekt z przedmiotu Algorytmy i struktury danych AGH 2008.
[10] Smutnicki C., *Algorytmy szeregowania*. Akademicka Oficyna Wydawnicza EXIT, Warszawa 2002.
[11] Szymczyk P.: *Systemy operacyjne czasu rzeczywistego*. UWND AGH, Kraków 2003.