Systemy czasu rzeczywistego

Mariusz Żynel
mariusz@math.uwb.edu.pl
http://math.uwb.edu.pl/~mariusz/

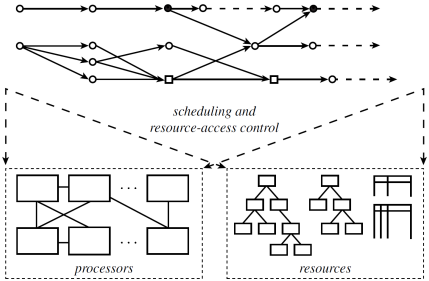
Uniwersytet w Białymstoku

2024/2025

Planowanie prac i harmonogramy

Model systemu czasu rzeczywistego

- Graf zadań określa wymagania prac co do czasu procesora i zasobów oraz ograniczenia czasowe i zależności pomiędzy pracami
- Graf zasobów opisuje ilość zasobów dostępnych do wykonania prac aplikacji, atrybuty zasobów i reguły regulujące ich wykorzystanie
- Pomiędzy grafem zadań i zasobów znajdują się algorytmy używane przez system operacyjny:
  - algorytmy planowania
  - algorytmy kontroli dostępu do zasobów



Model systemu czasu rzeczywistego

Planista

- *Planista* (*scheduler*) to moduł implementujący algorytmy planowania oraz protokoły kontroli dostępu do zasobów
- Planista przypisuje procesory do zadań (zadania do procesorów)
- Praca jest *zaplanowana* (*scheduled*) w przedziale czasu na procesorze, gdy procesor jest przypisany do tej pracy, czyli praca jest wykonywana na procesorze w tym przedziale
- Ilość czasu procesora przypisana do pracy to suma długości wszystkich takich przedziałów

- Dla uproszczenia zakładamy tutaj, że prace nigdy nie są wykonywane równolegle na więcej niż jednym procesorze

Harmonogram

- *Harmonogram* (*schedule*) to wygenerowane przez planistę przypisanie wszystkich prac w systemie do dostępnych procesorów
- Poprawny (valid) harmonogram spełnia następujące warunki:
  1. Każdy procesor jest przypisany do co najwyżej jednej pracy w dowolnym momencie
  2. Każda praca jest przypisana do co najwyżej jednego procesora w dowolnym momencie
  3. Żadna praca nie jest zaplanowane przed czasem jej uwolnienia
  4. W zależności od użytego algorytmu planowania, całkowita ilość czasu procesora przypisana do każdej pracy jest równa jej maksymalnemu lub rzeczywistemu czasowi wykonania
  5. Wszystkie zależności poprzedzania pomiędzy pracami oraz ograniczenia wykorzystania zasobów są spełnione

Wykonalność, optymalność i wydajność

- Harmonogram jest *wykonalny* (*feasible*), gdy jest poprawny i każda praca kończy się przed czasem ostatecznym (deadline), czyli spełnia ograniczenia czasowe
- Algorytm planowania jest *optymalny* (*optimal*), gdy zawsze generuje wykonalny harmonogram jeśli taki istnieje
- Ocena wydajności algorytmu dokonywana jest przy ustalonym kryterium
- Kryterium oceny wydajności zależy od tego co chcemy uzyskać
- Nie ma czegoś takiego jak wydajny algorytm, są tylko algorytmy wydajne ze względu na ...

Kryteria oceny wydajności algorytmów planowania

- Gdy nie istnieje harmonogram wykonalny możemy wybrać algorytm minimalizujący ilość prac spóźnionych, albo taki według którego nieważne ile prac jest spóźnionych byle sumaryczne spóźnienie było minimalne
- Czasem chcemy aby whania czasu ukończenia były jak najmniejsze (pakietowa tansmisja danych z buforem)
- Dla prac o miękich ograniczeniach czasowych, zwykłych systemów ogólnego przeznaczenia oraz systemów interaktywnych wskaźnikiem wydajności jest najczęściej średni czas reakcji, im krótszy tym algorytm jest wydajniejszy
- W systemach mieszanych, o pracach z twardymi i miekkimi ograniczeniami czasowymi, chcemy minimalizować średni czas rekacji dla prac miękkich oraz ukończyć wszystkie prace twarde na czas
- Ponieważ nie ma korzyści z wcześniejszego ukończenia twardych prac, możemy opóźnić ich wykonanie, aby poprawić czas reakcji prac miękkich

Wskaźniki chybień i strat

- W systemach miękkich dopuszczalne jest późniejsze ukończenie niektórych zadań lub odrzucenie opóźnionych zadań
- *Wskaźnik chybień* (*miss rate*) to procent zadań wykonywanych, ale ukończonych późno
- *Wskaźniki strat* (*loss rate*) to procent zadań odrzuconych, czyli w ogóle niewykonywanych
- Gdy nie można ukończyć wszystkich zadań na czas, planista może zdecydować się na odrzucenie niektórych zadań
- Zmniejszenie wskaźnika strat może prowadzić do wzrostu wskaźnika chybień
- Zmniejszenie wskaźnika chybień może prowadzić do wzrostu wskaźnika strat
- Minimalizacja wskaźnika chybień oznacza zmniejszenie tego wskaźnika tak bardzo jak to możliwe z zastrzeżeniem, że wskaźnik strat jest poniżej pewnego dopuszczalnego progu
- Minimalizacja wskaźnika strat oznacza zmniejszenie tego wskaźnika tak bardzo jak to możliwe z zastrzeżeniem, że wskaźnik chybień jest poniżej pewnego dopuszczalnego progu

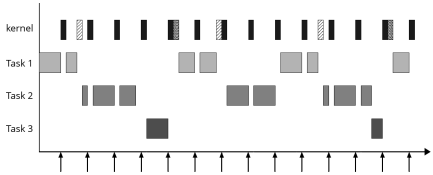
Strategie planowania w systemach czasu rzeczywistego

Podstawowe strategie

- Sterowane zegarem (czasem) (clock-driven, time-driven)
- Rotacyjne ważone (weighted round-robin)
- Oparte na priorytetach (priority-driven)

Planowanie sterowane zegarem (czasem)

- Decyzje dotyczące tego, które prace i kiedy mają być wykonywane, podejmowane są okresowo w określonych momentach
- Momenty te oraz harmonogram ustalane są off-line zanim system rozpocznie działanie
- Wszystkie parametry prac o twardych ograniczeniach czasowych są znane i ustalone
- Okresowe podejmowanie decyzji przez planistę realizowane jest za pomocą timera sprzętowego
- System działa w następujący sposób:
  - System jest inicjowany
  - Planista wybiera pracę do wykonania
  - Planista blokuje się
  - Wykonywana jest wybrana praca
  - Timer wygasa i przerwanie aktywuje planistę
  - Procedura jest powtarzana



Planowanie sterowane zegarem

Planowanie rotacyjne ważone

- W szeregowaniu według algorytmu *rotacyjnego* (*round-robin*), gdy praca jest gotowa do wykonania dołączana jest do kolejki FIFO (First-In-First-Out)
- Gdy praca dociera na szczyt kolejki wykonywana jest przez *kwant czasu* (*time slice*)
- Jeśli praca nie zostanie ukończona przed końcem kwantu czasu, zostaje wywłaszczona i umieszczona na końcu kolejki, aby poczekać na następną turę
- Ponieważ kwant czasowy jest mały (kilka milisekund) wykonanie każdej pracy rozpoczyna się niemal natychmiast po dodaniu jej do kolejki
- Każdej pracy przypisana może być pewna *waga* (*weight*)
- W każdej turze praca o wadze w wykonywana jest przez w kwantów czasu

- Czas reakcji łańcucha zależnych od siebie prac może być nadmiernie długi
- W przypadku zależności producent-konsument, gdy prace wykonywane są w potoku jedna za drugą, uzyskuje się dużą wydajność
- Szeregowanie rotacyjne ważone stosowane jest w sieciach o ultra wysokiej przepustowości

Planowanie oparte na priorytetach

- Zasób jest nieużywany tylko wtedy, gdy nie ma pracy gotowej do wykonania wymagającej tego zasobu
- Planista podejmuje decyzje w momencie, gdy mają miejsce zdarzenia takie jak: uwolnienie pracy lub ukończenie pracy
- Planowanie oparte na priorytetach jest sterowane zdarzeniami (event-driven)
- Pozostawienie zasobu nieużywanym, gdy pewna praca jest gotowa aby go użyć, nie jest lokalnie optymalne
- Algorytm oparty na priorytetach próbuje podejmować lokalnie optymalne decyzje
- Algorytm oparty na priorytetach jest algorytmem zachłannym

Priorytety, sposób działania

- Każdej pracy przypisany jest *priorytet* (*priority*)
- Prace gotowe do wykonania umieszczane są w jednej lub wielu kolejkach uporządkowanych według priorytetów
- W momencie podejmowania decyzji planista wybiera prace o najwyższym priorytecie do wykonania na dostępnym procesorze
- Większość algorytmów szeregowania używanych w systemach nie czasu rzeczywistego jest oparta na priorytetach:
  - Priorytety przypisywane zgodnie z czasami uwolnienia:
    - FIFO First-In-First-Out
    - LIFO Last-In-First-Out
  - Priorytety przypisywane zgodnie z czasami wykonania:
    - SETF Shortest-Execution-Time-First
    - LETF Longest-Execution-Time-First

```
> FIN < ACK < FIN > ACK

init 5
```