ЛЕКЦІЯ 4. ПЛАНУВАННЯ ПРОЦЕСІВ І ПОТОКІВ

План

4.1. Загальні принципи планування
4.2. Види планування
4.3. Стратегії планування
4.4. Алгоритми планування
4.5. Планування потоків і процесів в Windows XP

Процеси і потоки є активними ресурсами обчислювальних систем, які реалізують виконання програмного коду. *Потоком* називають набір послідовно виконуваних команд процесора. *Процес* є сукупністю одного або декількох потоків і захищеного адресного простору, в якому вони виконуються.

Потоки одного процесу можуть разом використовувати спільні дані, для потоків різних процесів без використання спеціальних засобів це неможливо.

4.1. Загальні принципи планування

4.1.1. Особливості виконання потоків

З погляду планування виконання потоку можна зобразити як цикл чергування періодів обчислень (використання процесора) і періодів очікування введення-виведення. Інтервал часу, упродовж якого потік виконує тільки інструкції процесора, називають *інтервалом використання процесора* (CPU burst), інтервал часу, коли потік очікує введення-виведення, - *інтервалом введення-виведення* (I/O burst). Найчастіше ці інтервали мають довжину від 2 до 8 мс.

Потоки, які більше часу витрачають на обчислення і менше — на введення-виведення, називають обмеженими можливостями процесора (CPU bound). Вони активно використовують процесор. Основною їхньою характеристикою є час, витрачений на обчислення, інтервали використання процесора для них довші. Потоки, які більшу частину часу перебувають в очікуванні введення-виведення, називають обмеженими можливостями введення-виведення (I/O bound). Такі потоки завантажують процесор значно менше, а середня довжина інтервалу використання процесора для них невелика. Що вища тактова частота процесора, то більше потоків можна віднести до другої категорії.

4.1.2. Механізми і політика планування

Слід розрізняти *механізми* і *політику* планування. До механізмів планування належать засоби перемикання контексту, засоби синхронізації потоків тощо, до політики планування - засоби визначення моменту часу, коли необхідно перемкнути контекст. Ту частину системи, яка відповідає за політику планування, називають планувальником (scheduler), а алгоритм, що використовують при цьому, - алгоритмом планування (scheduling algorithm).

Є різні критерії оцінки політики планування, одні з них застосовні для всіх систем, інші — лише для пакетних систем або лише для інтерактивних.

Сьогодні найчастіше використовують *три критерії оцінки досягнення мети.*

Мінімальний час відгуку. Це найважливіший критерій для інтерактивних систем. Під часом відгуку розуміють час між запуском потоку (або введенням користувачем інтерактивної команди) і отриманням першої відповіді. Для сучасних систем прийнятним часом відгуку вважають 50-150 мс.

Максимальна пропускна здатність. Це кількість задач, які система може виконувати за одиницю часу (наприклад, за секунду). Такий критерій доцільно застосовувати у пакетних системах; в інтерактивних системах він може бути використаний для фонових задач.

Третім критерієм є *справедливість*, який полягає в тому, що процесорний час потокам виділяють відповідно до їхньої важливості. Справедливість забезпечує такий розподіл процесорного часу, що всі потоки просуваються у своєму виконанні, і жоден не простоює.

4.1.3. Застосовність принципів планування

Принципи планування потоків застосовні насамперед до багатопотокових систем із реалізацією схеми 1:1 (тут плануються винятково потоки ядра), а також до систем з реалізацією моделі процесів. В останньому випадку замість терміна «потік» можна вживати термін «процес», а інформацію, необхідну для планування, зберігати в структурах даних процесів. Складніші принципи планування використовують у багатопотокових системах, для яких кількість потоків користувача не збігається з кількістю потоків ядра. Для них потрібні два планувальники: один для роботи на рівні ядра, інший — у режимі користувача.

4.2. Види планування

Розрізняють планування *довготермінове* (long-term scheduling), *середньотермінове* (medium-term scheduling) і *короткотермінове* (short-term scheduling).

Довготермінове планування

Засоби довготермінового планування визначають, яку з програм треба завантажити у пам'ять для виконання. Таке планування називають також статичним, оскільки воно не залежить від поточного стану системи. Використовується в пакетних системах.

Середньотермінове планування

Засоби середньотермінового планування керують переходом потоків із призупиненого стану в стан готовності й назад. Відразу ж зазначимо, що керуючі блоки готових до виконання потоків організуються у пам'яті в структуру, яку називають чергою готових потоків (ready queue).

Перехід потоку в призупинений стан можуть викликати такі фактори:

- очікування операції введення-виведення;
- очікування закінчення виконання іншого потоку (приєднання);
- блокування потоку через необхідність його синхронізації з іншими потоками.

Зазвичай для коректної організації такого очікування, крім черги готових потоків, реалізують додатковий набір черг. Кожна така черга пов'язана з ресурсом, який може викликати очікування потоку (наприклад, із пристроєм введення-виведення); ці черги ще називають чергами планування (scheduling queues) або чергами очікування (wait queues).

Короткотермінове планування

Короткотермінове планування, або планування процесора (CPU scheduling), є найважливішим видом планування. Воно дає змогу відповісти на два базових запитання.

1. Коли перервати виконання потоку?
2. Якому потокові з числа готових до виконання потрібно передати процесор у цей момент?

Короткотерміновий планувальник — це підсистема ОС, яка в разі необхідності перериває активний потік і вибирає з черги готових потоків той, що має виконуватися.

4.3. Стратегії планування

Усі стратегії й алгоритми планування, які ми будемо розглядати далі, належать до короткотермінового планування.

Перед тим як розглянути основні стратегії планування, перелічимо варіанти передачі керування від одного потоку до іншого:

- після того, як потік перейшов у стан очікування (наприклад, під час введення-виведення або приєднання);
- після закінчення виконання потоку;
- явно (потік сам віддає процесор іншим потокам на час, поки він не зайнятий корисною роботою);
- за перериванням (наприклад, переривання від таймера дає змогу перервати потік, що виконується довше, ніж йому дозволено).

Останній варіант відрізняється від інших тим, що потік не може контролювати, коли настане час передачі керування, за це відповідає планувальник операційної системи. Залежно від підтримки такого варіанта передачі керування розрізняють дві основні стратегії планування потоків — витісняльну і невитісняльну багатозадачність.

При *витісняльній багатозадачності* (preemptive multitasking) потоки, що логічно мають виконуватися, можуть бути тимчасово перервані планувальником ОС без їхньої участі для передачі керування іншим потокам. Переривання виконання потоку й передачу керування іншому потокові найчастіше здійснюють в обробнику переривання від системного таймера. Така стратегія реалізована в усіх сучасних ОС.

При *невитісняльній багатозадачності* (non-preemptive multitasking) потоки можуть виконуватися упродовж необмеженого часу й не можуть бути перервані ОС. Для невитісняльної багатозадачності передача керування за останнім варіантом не реалізована, і потоки самі повинні віддавати керування ОС для передачі іншим потокам або, принаймні, переходити у стан очікування.

4.4. Алгоритми планування

Як ми вже знаємо, алгоритм планування дає змогу короткотерміновому планувальникові вибирати з готових до виконання потоків той, котрий потрібно виконувати наступним. Можна сказати, що алгоритми планування реалізують політику планування.

Залежно від стратегії планування, яку реалізують алгоритми, їх поділяють на витісняльні та невитісняльні. Витісняльні алгоритми переривають потоки під час їхнього виконання, невитісняльні — не переривають.

4.4.1. Кругове планування

Найпростішим для розуміння і найсправедливішим витісняльним алгоритмом є алгоритм кругового планування (round-robin scheduling).

Кожному потокові виділяють інтервал часу, який називають квантом часу (time quantum, time slice) і упродовж якого цьому потокові дозволено виконуватися. Коли потік усе ще виконується після вичерпання кванта часу, його переривають і перемикають процесор на виконання інструкцій іншого потоку. Коли він блокується або закінчує своє виконання до вичерпання кванта часу, процесор теж передають іншому потокові. Довжина кванта часу для всієї системи однакова.

Такий алгоритм реалізувати досить легко. Для цього черга готових потоків має бути циклічним списком. Коли потік вичерпав свій квант часу, його переміщують у кінець списку, туди ж додають і нові потоки (рис. 4.1).

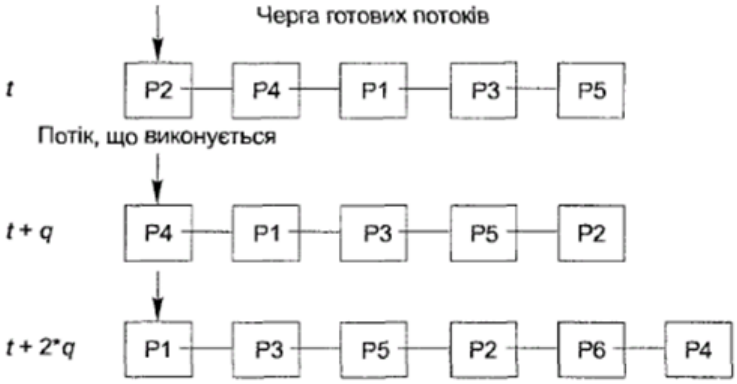

Рис. 4.1 Кругове планування

Перевірку вичерпання кванта часу виконують в обробнику переривання від системного таймера

Єдиною характеристикою, яка впливає на роботу алгоритму, є довжина кванта часу. Тут слід дотримуватися балансу між часом, що витрачається на перемикання контексту, і необхідністю відповідати на багато одночасних інтерактивних запитів.

4.4.2. Планування із пріоритетами

Планування за принципом кругового чергування припускає, що всі потоки однаково важливі. В іншому разі необхідно застосовувати планування із пріоритетами. Основна ідея проста: кожному потокові надають пріоритет, при цьому на виконання ставитиметься потік із найвищим пріоритетом із черги готових потоків. Пріоритети можуть надаватися потокам статично або динамічно.

Одним із підходів до реалізації планування із пріоритетами є алгоритм багаторівневих черг (multilevel queues). У цьому разі організовують кілька черг для груп потоків із різними пріоритетами (потоки кожної групи звичайно мають різне призначення, можуть бути групи фонових потоків, інтерактивних тощо).

Рішення про вибір потоку для виконання приймають таким чином:

- якщо в черзі потоків із найвищим пріоритетом є потоки, для них слід використати якийсь простіший алгоритм планування (наприклад, кругового планування), не звертаючи уваги на потоки в інших чергах;

- якщо в черзі немає жодного потоку, переходять до черги потоків з нижчим пріоритетом і т. д.

Для різних черг можна використовувати різні алгоритми планування, крім того, кожній черзі може бути виділена певна частка процесорного часу.

4.4.3. Планування на підставі характеристик подальшого виконання

Важливим класом алгоритмів планування з пріоритетами є алгоритми, в яких рішення про вибір потоку для виконання приймають на підставі знання або оцінки характеристик подальшого його виконання.

Насамперед слід відзначити алгоритм «перший — із найкоротшим часом виконання» (Shortest Time to Completion First, STCF), коли з кожним потоком пов'язують тривалість наступного інтервалу використання ним процесора і для виконання щоразу вибирають той потік, у якого цей інтервал найкоротший. У результаті потоки, що захоплюють процесор на коротший час, отримують під час планування перевагу і швидше виходять із системи. Алгоритм STCF є теоретично оптимальним за критерієм середнього часу відгуку, тобто можна довести, що для вибраної групи потоків середній час відгуку в разі застосування цього алгоритму буде мінімальним порівняно з будь-яким іншим алгоритмом.

Витісняльним аналогом STCF є алгоритм «перший — із найкоротшим часом виконання, що залишився» (Shortest Remaining Time to Completion First, SRTCF). Його відмінність від SCTF полягає в тому, що, коли в чергу готових потоків додають новий, у якого наступний інтервал використання

процесора коротший, ніж час, що залишився до завершення виконання поточного потоку, поточний потік переривається, і на його місце стає новий потік.

4.4.4. Багаторівневі черги зі зворотним зв'язком

Алгоритм багаторівневих черг зі зворотним зв'язком (multilevel feedback queues) є найбільш універсальним алгоритмом планування, але при цьому одним із найскладніших у реалізації.

З погляду організації структур даних цей алгоритм схожий на звичайний алгоритм багаторівневих черг: є кілька черг готових потоків із різним пріоритетом, при цьому потоки черги із нижчим пріоритетом виконуються, тільки коли всі черги верхнього рівня порожні.

Відмінності між двома алгоритмами полягають у тому, що:

- потокам дозволено переходити з рівня на рівень (із черги в чергу);
- потоки в одній черзі об'єднуються не за пріоритетом, а за довжиною інтервалу використання процесора, потоки із коротшим інтервалом перебувають у черзі з більшим пріоритетом.

Усередині всіх черг, крім найнижчої, використовують кругове планування. Різні черги відповідають різній довжині кванта часу — що вищий пріоритет, то коротший квант (звичайно довжина кванта для сусідніх черг зменшується удвічі). Якщо потік вичерпав свій квант часу, він переміщається у хвіст черги із нижчим пріоритетом (і з довшим квантом). У результаті потоки з коротшими інтервалами (наприклад, обмежені введенням-виведенням) залишаються з високим пріоритетом, а потоки з довшими інтервалами подовжують свій квант часу.

4.4.5. Лотерейне планування

Лотерейне планування (lottery scheduling) — це останній алгоритм, який ми розглянемо. Він простий для розуміння і легкий у реалізації, разом з тим має великі можливості.

Ідея лотерейного планування полягає у тому, що:

- потік отримує деяку кількість лотерейних квитків, кожен з яких дає право користуватися процесором упродовж часу T,
- планувальник через проміжок часу T вибирає випадково один лотерейний квиток;
- потік, «що виграв», дістає керування до наступного розіграшу.

Лотерейне планування дає змогу:

- емулювати кругове планування, видавши кожному потокові однакову кількість квитків;
- емулювати планування із пріоритетами, розподіляючи квитки відповідно до пріоритетів потоків;
- забезпечити розподіл процесорного часу між потоками — дати кожному потокові кількість квитків, пропорційну до частки процесорного часу, який потрібно йому виділити.

Хоча більшість цих задач може бути розв'язана лотерейним плануванням лише приблизно, з деякою ймовірністю, на практиці отримують цілком задовільні результати. При цьому що довше працює система, то ближчими будуть результати до теоретичних значень (за законом великих чисел).

4.6. Реалізація планування у Windows XP

4.6.1. Планування потоків у ядрі

Ядро Windows XP розв'язує під час планування дві основні задачі:

- облік відносних пріоритетів, присвоєних кожному потокові;
- мінімізацію часу відгуку інтерактивних застосувань.

Базовою одиницею планування є потік. Під час планування ядро не розрізняє потоки різних процесів, воно має справу з пріоритетами потоків, готових до виконання в певний момент часу.

Під час планування ядро працює з мінімальними версіями потоків. У них зберігається така інформація, як загальний час виконання потоку, його базовий і поточний пріоритет, диспетчерський стан потоку (готовність, очікування, виконання тощо).

Пріоритети потоків і процесів

Для визначення порядку виконання потоків диспетчер ядра використовує систему пріоритетів. Кожному потокові присвоюють пріоритет, заданий числом у діапазоні від 1 до 31 (що більше число, то вище пріоритет). Пріоритети реального часу — 16-31; їх резервує система для дій, час виконання яких є критичним чинником. Динамічні пріоритети — 1-15; вони можуть бути присвоєні потокам застосувань користувача.

Ядро системи може надати потоку будь-який динамічний пріоритет. Win32 API не дає можливості зробити це з цілковитою точністю, у ньому використовують дворівневу систему, яка зачіпає як процес, так і його потоки: спочатку процесу присвоюють клас пріоритету, а потім потокам цього процесу — відносний пріоритет, який відраховують від класу пріоритету процесу.

Розрізняють такі класи пріоритету процесів: реального часу (real-time, приблизно відповідає пріоритету потоку 24); високий (high, 13); нормальний (normal, 8); невикористовуваний (idle, 4).

Відносні пріоритети потоку бувають такі: найвищий (+2 до базового); вище за нормальний (+1 до базового); нормальний (дорівнює базовому); нижче за нормальний (-1 від базового); найнижчий (-2 від базового).

Є два додаткових модифікатори відносного пріоритету: критичний за часом (time-critical) і невикористовуваний (idle). Перший модифікатор тимчасово задає для потоку пріоритет 15 (найвищий динамічний пріоритет), другий аналогічним чином задає пріоритет 1.

Особливості задання кванту часу

Важливою характеристикою системи є довжина кванта часу. Розрізняють короткі й довгі кванти, для яких можна задати змінну та фіксовану довжину.

У Windows XP інтерактивно можна задавати таку довжину кванта (вибирають Settings (Параметры) у групі Performance (Быстродействие) на вкладці Advanced (Дополнительно) вікна властивостей My Computer (Свойства системы)):

- короткі кванти змінної довжини (вкладка Advanced (Дополнительно), перемикач Programs (Программ) у групі властивостей Processor Scheduling (Распределение времени процесора)). Можлива довжина кванта - 10 або 30мс, при цьому застосування, з яким починає працювати користувач, автоматично переходить до використання довших квантів. Ця установка надає перевагу інтерактивним процесам;

- довгі кванти фіксованої довжини (вкладка Advanced (Дополнительно), перемикач Background services (Служб, работающих в фоновом режиме) у групі властивостей Processor Scheduling (Распределение времени процесора)). Довжина кванта фіксована й дорівнює 120 мс. Ця установка надає перевагу фоновим процесам.

Пошук потоку для виконання

Для виконання новий потік вибирається, коли:

- минув квант часу для потоку (з використанням алгоритму пошуку готового потоку);

- потік перейшов у стан очікування події (потік сам віддає квант часу і дає команду планувальникові запустити алгоритм пошуку готового потоку);

- потік перейшов у стан готовності до виконання (використовують алгоритм розміщення готового потоку).

Планувальник підтримує спеціальну структуру даних — список готових потоків (dispatcher ready list). У цьому списку зберігається 31 елемент — по одному для кожного рівня пріоритету. З кожним елементом пов'язана черга готових потоків, всі потоки з однаковим пріоритетом перебувають у черзі, яка відповідає їхньому рівню пріоритету.

Під час виконання алгоритму пошуку готового потоку планувальник переглядає всі черги потоків, починаючи з черги найвищого пріоритету. Як тільки під час цього перегляду трапляється потік, його відразу вибирають для виконання. За допомогою цього алгоритму вибирають перший потік непустої черги з найвищим пріоритетом.

Якщо подивитися на ситуацію з боку потоку, що виконується, то важливо знати, коли він може бути витіснений. Це трапляється коли:

- потік перейшов у стан очікування;

- минув квант часу потоку;

- потік із вищим пріоритетом перейшов у стан готовності до виконання;

- змінився пріоритет потоку або пріоритет іншого потоку.

Динамічна зміна пріоритету і кванту часу

Під час виконання потоків динамічний пріоритет і довжина кванта часу можуть бути скориговані ядром системи. Розрізняють два види такої динамічної зміни: *підтримка* (boosting) і *ослаблення* (decay).

Підтримка зводиться зазвичай до тимчасового підвищення пріоритету потоків. Коли потік переходить у стан готовності до виконання внаслідок настання події, на яку він очікував, виконують операцію підтримки.

Під час завершення операції введення-виведення підвищення пріоритету залежить від типу операції. Наприклад, після виконання дискових операцій пріоритет збільшують на одиницю, після введення із клавіатури або обробки події від миші — на 6.

Під час зміни стану синхронізаційного об'єкта пріоритет потоку, який очікує цієї зміни, збільшують на одиницю.

Вихід з будь-якого стану очікування для потоків інтерактивних застосувань призводить до підвищення пріоритету на 2, таке саме підвищення відбувається під час переходу в стан готовності потоків, пов'язаних із відображенням інтерфейсу користувача.

Зазначимо, що внаслідок операцій підтримки динамічний пріоритет потоку не може перевищити значення 15 (максимально допустимого динамічного пріоритету). Якщо операція підтримки вимагає підвищення пріоритету до величини, вищої за це значення, пріоритет збільшують тільки до 15.

Підвищення пріоритету внаслідок *підтримки* дедалі слабшає. Після закінчення кожного кванта часу поточний пріоритет потоку зменшують на одиницю, поки він не дійде до базового, після чого пріоритет залишають на одному рівні до наступної операції підтримки.

Ще одним видом підтримки є зміна кванта часу для інтерактивних застосувань. Якщо під час налаштування системи задано використання квантів змінної довжини, можна вказати, що для інтерактивних застосувань довжина кванта буде збільшуватися (це називають підтримкою кванта для інтерактивних застосувань). Якщо така підтримка задана, то коли інтерактивне застосування захоплює фокус, всі його потоки отримують квант часу, який дорівнює значенню підтримки (дозволене одне з можливих значень кванта, наприклад, 40 або 60 мс).

З іншого боку, значення кванта може й зменшуватися (слабшати). Так, під час виконання будь-якої функції очікування довжина кванта зменшується на одиницю.

Для потоків із пріоритетом реального часу динамічна зміна пріоритету або довжини кванта ніколи не відбувається. Єдиний спосіб змінити пріоритет таких потоків — викликати відповідну функцію із прикладної програми.

Запобігання голодуванню

Якщо в системі постійно є потоки з високим пріоритетом, може виникати голодування потоків, пріоритет яких нижчий. Для того щоб уникнути голодування, спеціальний потік ядра один раз за секунду обходить чергу готових потоків у пошуках тих, які перебували у стані готовності досить довго (понад 3 с) і жодного разу не отримали шансу на виконання. Коли такий потік знайдено, то йому присвоюють пріоритет 15 (і він дістає змогу негайного виконання); крім того, довжину його кванта часу подвоюють. Після того, як два кванти часу минають, пріоритет потоку і його квант повертаються до вихідних значень.

Цей алгоритм не враховує причин голодування і не розрізняє потоків інтерактивних і фонових процесів.

Висновки

1. Задача планування потоків зводиться до організації виконання кількох потоків на одному процесорі так, аби у користувачів виникало враження, що вони виконуються одночасно. Цілями планування є: мінімізація часу відгуку, максимізація пропускної здатності та справедливість. До основних стратегій планування належать витісняльна й невитісняльна багатозадачність. У сучасних ОС застосовують *витісняльну багатозадачність*, коли рішення про перемикання контексту потоку приймають у коді ядра системи, а не в коді потоку.

2 Розрізняють довготермінове, середньотермінове й короткотермінове планування. Найважливіший тут короткотерміновий планувальник, котрий використовують для прийняття рішення про те, який потік запустити на виконання в певний момент. До основних алгоритмів короткотермінового планування належать планування *кругове* і з *пріоритетами*.

Контрольні запитання і завдання

1. Назвіть основні цілі та стратегії планування процесів і потоків.
2. Опишіть алгоритм планування із пріоритетами.
3. Які переваги дає реалізація алгоритму лотерейного планування потоків?
4. Опишіть алгоритм кругового планування. У чому полягають його переваги порівняно з іншими алгоритмами планування?
5. Яку стратегію планування використано в алгоритмі кругового планування: витісняльну чи ні?
6. У чому різниця між механізмами і політикою планування потоків і процесів? Що таке "Планувальник" і за що він відповідає?
7. Які є критерії оцінки політики планування? Чим вони відрізняються між собою?
8. Що означає "мінімальний час відгуку"?
9. Де і для чого застосовуються принципи планування потоків?
10. У чому полягає "голодування потоків"? Яким чином ОС виконує запобігання голодуванню?

Література до лекції 4.

1. Шеховцов В. А. Операційні системи. Підручник для ВНЗ. – К.: BHV, 2008. –576 с.
2. Таненбаум Э. Операционные системы– СПб: Питер. – 2002 г. – 1040 с.