12.1 ПЛАНУВАННЯ В СИСТЕМАХ З ОДНИМ ПРОЦЕСОРОМ

За часів пакетної обробки, алгоритм планування був простий: запустити певну задачу зі стрічки або з перфокарт. З появою систем розподілу часу алгоритм планування ускладнився, оскільки тепер декілька задач очікували обслуговування. На деяких мейнфреймах досі поєднуються системи пакетної обробки, і служби розподілу часу. У результаті планувальник повинен вирішувати запускати наступне пакетне завдання чи надати процесор інтерактивному завданню.

З появою ПК ситуація змінилася. По-перше, більшу частину часу активний тільки один процес. По-друге, ПК стали настільки швидкими, що час процесора вже став не таким дефіцитним ресурсом. Більшість програм обмежені швидкістю, з якою користувач вводить вхідні дані (з клавіатури або за допомогою миші), а не швидкістю процесора. Навіть процедури компіляції, основні споживачі процесорного часу, тепер займають кілька секунд.

Картина також змінилася, коли появились потужні робочі станції і сервери. Тут планування знову грає істотну роль, оскільки кілька процесів намагаються отримати доступ до ресурсу. Наприклад, коли процесору потрібно вибрати між процесом перемальовування екрану після того, як користувач закрив вікно додатку, і процесом, що відсилає пошту, враження користувача від реакції комп'ютера буде істотно залежати від цього вибору. Адже, якщо перемальовування екрану під час відправки пошти займе 2 секунди, то користувач вирішить, що система дуже повільна, тоді як 2-х секундне відсилання пошти навіть не помітять. У цьому випадку планування процесів дуже важливе. Можливість паралельного виконання процесів (потоків) залежить від кількості доступних процесорів. Якщо процесор один, паралельне виконання неможливе (в кожен момент часу може виконуватися тільки один процес). Якщо процесорів $N > 1$, паралельне виконання може бути реалізовано для N процесів (потоків, по одному на процесор).

У багатозадачних системах в основний пам'яті одночасно міститься код декількох процесів. У роботі кожного процесу періоди використання процесора чергуються з очікуванням завершення виконання операцій введення-виведення або

деяких зовнішніх подій. Процесор зайнятий виконанням одного процесу, в той час як інші перебувають в стані очікування.

ОС повинна вирішувати **задачу планування** (**scheduling**), основна мета якої для однопроцесорної системи полягає в такій організації виконання процесів, через яку у користувача системи виникає враження, що вони виконуються одночасно.

Для організації управління процесами необхідно врахувати щонайменше два основних аспекти: визначення рівня, на якому виконується планування процесів і вибір алгоритму планування.

Таким чином, ключем до багатозадачності є планування. Як правило, використовуються три види (рівні) планування:

1. Довгострокове планування (планування верхнього рівня).
2. Середньострокове планування (планування проміжного рівня).
3. Короткострокове планування (планування нижнього рівня).

12.1.1 Рівні планування процесів

Однією з важливих задач, яку вирішує ОС, є проблема, пов'язана з визначенням коли і яким процесам слід виділяти ресурси процесора – задача планування завантаження процесора. Планування впливає на продуктивність системи, оскільки саме воно визначає, який процес буде працювати, а який повинен буде очікувати на виконання. Існують три рівні такого планування (рис. 12.1).

Рисунок 12.1 – Рівні планування

Довгострокове планування. Довгострокове планування (або планування завдань) вказує, які програми (завдання) допускаються до виконання системою і тим самим визначає ступінь багатозадачності. Будучи допущеним до виконання, завдання стає процесом, який додається в чергу для короткострокового планування. У деяких системах новостворений процес додається до черги середньострокового планування, будучи цілком скинутим на диск.

Планування завдань з'явилося в пакетних системах після того, як для зберігання сформованих пакетів завдань почали використовуватися магнітні диски. Магнітні диски, будучи пристроями прямого доступу, дозволяють завантажувати завдання в комп'ютер у довільному порядку, а не тільки в тому, в якому вони були записані на диск. Змінюючи порядок завантаження завдань в обчислювальну систему, можна підвищити ефективність її використання.

Рішення про те, коли треба створити новий процес, визначаються загальним рівнем багатозадачності. Чим більше процесів буде створено, тим менший відсоток часу витрачатиметься на виконання кожного з них, але повніше буде завантажений процесор. Кожен раз при завершенні завдання довгостроковий планувальник вирішує, чи треба додавати в систему один або кілька нових процесів. Крім того, довгостроковий планувальник може бути викликаний в разі, коли відносний час простою процесора перевищує певний поріг.

Рішення про те, яке з завдань має бути додано в систему, може ґрунтуватися на простому принципі «першим прийшов – першим обслуговується». Крім того, для управління продуктивністю системи може використовуватися і спеціальний інструментарій, який враховує пріоритет задач, час очікуваного виконання і вимоги для роботи пристроїв введення-виведення.

У разі використання інтерактивних програм в системах з розподіленням часу запит на запуск процесу може здійснюватися внаслідок обслуговування підключення до системи. ОС приймає всіх зареєстрованих користувачів до насичення системи (поріг визначається заздалегідь). Після досягнення системою стану насичення на всі запити на вхід в систему, буде отримано повідомлення про заповнення системи і тимчасово припинення доступу до неї з пропозицією повторити операцію входу пізніше.

Якщо ступінь мультипрограмування системи підтримується постійною, тобто середня кількість процесів у комп'ютері не змінюється, то нові процеси можуть з'являтися тільки після завершення раніше завантажених. Тому довгострокове планування здійснюється досить нечасто, між появою нових процесів можуть проходити хвилини і навіть десятки хвилин. Звідси і назва цього рівня планування – довгострокове.

Середньострокове планування. У деяких обчислювальних системах буває вигідно для підвищення їх продуктивності тимчасово видалити будь-який процес, який виконується, з оперативної пам'яті на диск, а пізніше повернути його назад для подальшого виконання. Така процедура в англомовній літературі отримала назву swapping, що можна перевести як перекачування (підкачка), хоча у фаховій літературі воно вживається без перекладу – свопінг. Коли і який із процесів потрібно перекачати на диск і повернути назад, вирішується додатковим проміжним рівнем планування процесів – середньостроковим.

Рішення про завантаження процесу в пам'ять приймається в залежності від ступеня багатозадачності. Крім того, в системі з відсутністю віртуальної пам'яті середньострокове планування також тісно пов'язане з питаннями управління пам'яттю. Таким чином, рішення про завантаження процесу в пам'ять має враховувати вимоги до пам'яті вивантажуваного процесу.

Середньострокове планування керує переходом процесів з призупинених станів в стан готовності і назад. Керуючі блоки готових до виконання процесів організовуються в пам'яті в структуру, яку називають чергою готових процесів. Детальніше розглянемо ці черги під час короткострокового планування.

Короткострокове планування. Короткострокове планування (диспетчеризація або планування процесора) є найважливішим видом планування. Розглядаючи частоту роботи планувальника, можна сказати, що довгострокове планування виконується відносно нечасто, середньострокове – дещо частіше. Короткостроковий планувальник, відомий також як диспетчер, працює найчастіше, визначаючи, який саме процес буде виконуватися наступним. Вибір нового процесу для виконання впливає на функціонування системи до настання чергової аналогічної події.

Короткостроковий планувальник викликається при настанні події, що може призупинити поточний процес або надати можливість припинити виконання даного процесу на користь іншого. Приклади таких подій: переривання таймера, переривання введення-виведення, виклики ОС, сигнали.

Всі стратегії і алгоритми планування, які ми будемо розглядати далі, належать до короткострокового планування (диспетчеризації).

Місце планування в графі станів і переходів процесів показано на рис. 12.2.

Рисунок 12.2 – Місце планування в графі процесів

12.1.2 Параметри планування

Для здійснення поставлених цілей розумні алгоритми планування повинні спиратися на будь-які характеристики процесів у системі, завдань в черзі на завантаження, стану самої обчислювальної системи, іншими словами, на параметри планування. У цьому розділі ми опишемо ряд таких параметрів, не претендуючи на повноту викладу.

Усі параметри планування можна розбити на дві великі групи: статичні і динамічні. Статичні параметри не змінюються в ході функціонування обчислювальної системи, динамічні ж, навпаки, схильні до постійних змін.

До статичних параметрів обчислювальної системи можна віднести граничні значення її ресурсів вже на етапі завантаження (розмір оперативної пам'яті, максимальна кількість пам'яті на диску для здійснення свопінгу, кількість підключених пристроїв введення-виведення тощо).

1. Яким користувачем запущений процес або сформовано завдання.

2. Наскільки важливою є поставлена задача, тобто, її пріоритет.

3. Скільки процесорного часу запрошено користувачем для задачі.

4. Яке співвідношення процесорного часу і часу, необхідного для здійснення операцій введення-виведення.

5. Які ресурси обчислювальної системи (оперативна пам'ять, пристрої введення-виведення, спеціальні бібліотеки, системні програми тощо) і в якій кількості необхідні завданню.

Динамічні параметри системи описують кількість вільних ресурсів у поточний момент часу.

Алгоритми довгострокового планування використовують у своїй роботі статичні і динамічні параметри обчислювальної системи і статичні параметри процесів (динамічні параметри процесів на етапі завантаження завдань ще не відомі). Алгоритми короткострокового і середньострокового планування додатково враховують і динамічні характеристики процесів. Для середньострокового планування в якості таких характеристик, наприклад, може виступати інформація.

1. Скільки часу пройшло з моменту вивантаження процесу на диск або його завантаження в оперативну пам'ять.

2. Скільки оперативної пам'яті займає процес.

3. Скільки процесорного часу вже було надано процесу.

Для короткострокового планування нам знадобиться ввести ще два динамічних параметри. Діяльність будь-якого процесу можна представити як послідовність циклів використання процесора і очікування завершення операцій введення-виведення. Проміжок часу безперервного використання процесора носить

англійською мовою назву CPU burst, а проміжок часу безперервного очікування введення-виведення – I/O burst. На рис. 12.3 показаний фрагмент діяльності деякого процесу на псевдомові програмування з виділенням вказаних проміжків. Для стислості викладу ми будемо використовувати терміни CPU burst і I/O burst без перекладу. Значення тривалості останніх і чергових CPU burst і I/O burst є важливими динамічними параметрами процесу.

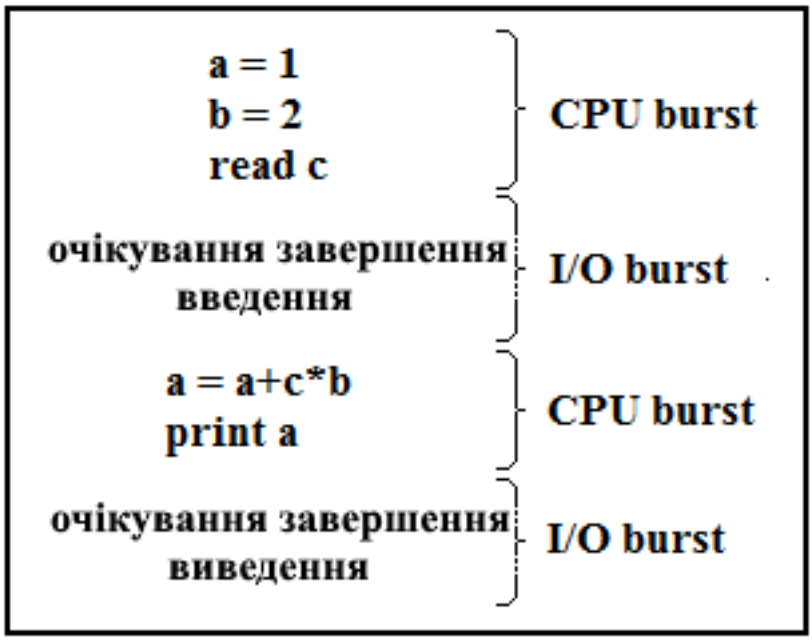

Рисунок 12.3 – Фрагмент діяльності процесу з виділенням проміжків безперервного використання процесора і очікування введення-виведення

12.1.3 Алгоритми планування

Основна мета короткострокового планування полягає в розподілі процесорного часу таким чином, щоб оптимізувати один або кілька аспектів поведінки системи. Існує певна кількість критеріїв оцінки різних стратегій планування. Найпоширеніші критерії можуть бути класифіковані в двох площинах.

По-перше, розділимо їх на призначені для користувача і системні. Призначені для користувача критерії пов'язані з поведінкою системи стосовно окремого користувача або процесу. Як приклад можна навести час відгуку в інтерактивній системі – це інтервал часу між передачею запиту і початком відповіді на нього. Його користувач відчуває безпосередньо. Для забезпечення якісного сервісу для користувачів треба встановити поріг відгуку, наприклад, 2 с. Тоді мета механізму

планування повинна складатися в максимізації кількості користувачів, середній час відгуку для яких не перевищує 2 с.

Системні критерії орієнтовані на ефективність і повноту використання процесора (пропускна здатність). Як приклад можна навести пропускну здатність, яка представляє собою швидкість завершення процесів. Це, безумовно, ефективна міра продуктивності системи, яка повинна бути максимальною. Однак вона більшою мірою орієнтована на продуктивність системи, а не на обслуговування користувача. Критерії, які орієнтовані на продуктивність, можуть бути виражені чисельними значеннями, і легко вимірюватися.

З іншого боку, призначені для користувача критерії важливі майже для всіх систем, системні критерії для систем одного користувача не так значимі. В цьому випадку, мабуть, досягнення високої ефективності використання процесора або висока продуктивність не так істотні, як швидкість відповіді системи додатку користувача. Назвемо ключові критерії планування.

Критерії, які призначені для користувача і пов'язані з продуктивністю:

1. Час обороту – інтервал часу між подачею процесу і його завершенням. Включає час виконання, а також час на очікування ресурсів, у тому числі і процесора (для пакетних задач).

2. Час відгуку – в інтерактивних процесах це час, який минув між поданням запиту і початком отримання відповіді на нього. Стратегія планування повинна спробувати скоротити час отримання відповіді при максимізації кількості інтерактивних користувачів, час відгуку для яких не виходить за задані межі.

3. Граничний термін – у разі зазначення граничного терміну завершення процесу планування має підпорядкувати йому всі інші цілі максимізації кількості процесів, що завершуються в указаний термін.

Критерії, які призначені для користувача, інші:

Передбачуваність – певна задача має виконуватися за один і той же час незалежно від завантаження системи. Великі зміни часу виконання або часу відгуку дезорієнтують користувачів. Це може сигналізувати про велику завантаженість ОС або про необхідність додаткового налаштування системи для усунення нестабільності її роботи.

Системні критерії, пов'язані з продуктивністю:

1. Пропускна здатність – стратегія планування повинна намагатися максимізувати кількість процесів, що завершуються за одиницю часу. Це значення залежить від середньої тривалості процесу, але при цьому на нього впливає і використовувана стратегія планування.

2. Використання процесора – це відсоток часу, коли процесор зайнятий.

Системні, інші критерії:

1. Неупередженість – при відсутності додаткових вказівок від користувача або системи всі процеси повинні розглядатися як рівнозначні і жоден з них не повинен «голодувати».

2. Використання пріоритетів – якщо процесам призначені пріоритети, то стратегія планування повинна віддавати перевагу процесам з більш вищим пріоритетом.

3. Баланс ресурсів – стратегія планування повинна підтримувати зайнятість системних ресурсів. Перевагу треба надати процесу, який недостатньо використовує важливі ресурси. Цей пріоритет включає використання довгострокового і середньострокового планування.

Всі ці критерії планування взаємозалежні, і досягти оптимального результату по кожному з них одночасно неможливо. Наприклад, забезпечення хорошого відгуку може зажадати застосування алгоритму з високою частотою перемикання процесів, що підвищить накладні витрати і, отже, знизить пропускну здатність системи. Отже, розробка стратегії планування являє собою пошук компромісу серед суперечливих вимог.

Розрізняють дві основні стратегії планування – **витісняючу** і **невитісняючу багатозадачність**. При витісняючій багатозадачності процеси, які виконуються, можуть бути перервані планувальником ОС без їх участі для передачі управління іншому процесу. Найчастіше це здійснюється оброблювачем переривань від системного таймера. Така стратегія реалізована в усіх сучасних ОС.

При невитісняючій багатозадачності процес може виконуватися протягом необмеженого часу і не може бути перерваний ОС. Процеси самі віддають управління

ОС, або переходити в стан очікування. Така стратегія була реалізована в MS Windows 3.1 і ОС Novell Net Ware 3.11 в 90-і роки.

Основною відмінністю між витісняючими і невитісняючими алгоритмами є ступінь централізації механізму планування процесів (потоків). При витісняючому мультипрограмуванні функції планування процесів цілком зосереджені в операційній системі. Програміст пише свій додаток, не піклуючись про те, що він буде виконуватися одночасно з іншими задачами. При цьому операційна система виконує такі функції: визначає момент зняття з виконання активного процесу, запам'ятовує його контекст, вибирає з черги готових процесів наступний, запускає новий процес на виконання, завантажуючи його контекст.

При невитісняючому мультипрограмуванні механізм планування розподілений між операційною системою і прикладними програмами.

Прикладна програма, отримавши управління від операційної системи, сама визначає момент завершення чергового циклу свого виконання і тільки потім передає керування ОС за допомогою будь-якого системного виклику. ОС формує черги процесів і вибирає відповідно до деякого правила (наприклад, з урахуванням пріоритетів) наступний процес на виконання. Такий механізм створює проблеми як для користувачів, так і для розробників додатків.

Для користувачів це означає, що управління системою втрачається на довільний період часу, який визначається додатком (а не користувачем). Якщо додаток витрачає занадто багато часу на виконання будь-якої роботи, наприклад на форматування диска, користувач не може переключитися з задачі на іншу задачу, наприклад, на текстовий редактор, в той час як форматування тривало б у фоновому режимі.

Тому розробники додатків для операційного середовища з витісняючою багатозадачністю змушені, покладаючи на себе частину функцій планувальника, створювати додатки так, щоб вони виконували свої задачі невеликими частинами. Наприклад, програма форматування може відформатувати одну доріжку дискети і повернути управління системі. Після виконання інших задач система поверне управління програмою форматування, щоб та відформатувала наступну доріжку.

Подібний метод поділу часу між задачами працює, але він істотно ускладнює розробку програм і висуває підвищені вимоги до кваліфікації програміста. Програміст повинен забезпечити «дружнє» ставлення своєї програми до інших виконуваних одночасно з нею програм. Для цього в програмі повинні бути передбачені часті передачі управління операційній системі. Крайнім виявом «не дружелюбності» додатка є його зависання, яке призводить до загального краху системи. У системах з витіснячою багатозадачністю такі ситуації, як правило, виключені, так як центральний планувальний механізм має можливість зняти зависле завдання з виконання.

Однак розподіл функцій планування потоків між системою і додатками не завжди є недоліком, а за певних умов може бути і перевагою, тому що дає можливість розробнику додатків самому проектувати алгоритм планування, найбільш підходящий для даного фіксованого набору завдань. Так як розробник сам визначає в програмі момент повернення управління, то при цьому виключаються нераціональні переривання програм в «незручні» для них моменти часу.

Крім того, легко вирішуються проблеми спільного використання даних: завдання під час кожного циклу виконання використовує їх монопольно і впевнено, що протягом цього періоду ніхто інший не змінить дані. Істотною перевагою невитісняючого планування є вища швидкість перемикання з процесу (потоку) на процес.

Майже в усіх сучасних ОС, орієнтованих на високу продуктивність виконання додатків (UNIX, Windows NT/2000/XP, OS/2), реалізовані витісняючі алгоритми планування потоків (процесів). Прикладом ефективного використання невитісняючого планування є файл-сервери NetWare 3.x/4.x, в яких досягнута висока швидкість виконання файлових операцій.