

Преамбула

Спочатку йдуть білети - просто питання. У третьому розділі - відповіді. До вирішених питань прикріплені теги. Для тегування використовувалися асоціації, англійські і транслітеровані скорочення. Шукайте найкоротші слова. А ще краще пробіжіться перед екзаменом по тегах і питаннях. Дайте мозку вловити мої асоціації. :)

Білети - лінки на відповіді

[Білет 1](#)
[Білет 2](#)
[Білет 3](#)
[Білет 4](#)
[Білет 5](#)
[Білет 6](#)
[Білет 7](#)
[Білет 8](#)
[Білет 9](#)
[Білет 10](#)
[Білет 11](#)
[Білет 12](#)
[Білет 13](#)
[Білет 14](#)
[Білет 15](#)
[Білет 16](#)
[Білет 17](#)
[Білет 18](#)
[Білет 19](#)
[Білет 20](#)
[Білет 21](#)
[Білет 22](#)
[Білет 23](#)
[Білет 24](#)
[Білет 25](#)
[Білет 26](#)
[Білет 27](#)
[Білет 28](#)

Білети

Білет 1

1. У чому сенс закону Гроша й чому він зараз не діє?
2. На що впливає введення поняття «надійний стан системи»?
3. Умови створення процесу. Хто створює процес.
4. Чому P3 і P4 не мають циклу?
5. Види завантажників. Їх основні відмінності.
6. Дати визначення ініціалізації системи. Мета ініціалізації.
7. Дати визначення програми простої структури. Переваги й недоліки.
8. Функції редактора зв'язків. Звідки він бере інформацію для зв'язування?
9. Дати визначення статичного планування.
10. Що таке критичний шлях та критичний час?
11. Що таке перегляд команд вперед та яка характеристика при цьому покращується?
12. Що таке 'свопінг'?
13. Перерахувати стратегії, що застосовуються в алгоритмах оптимізації.
14. Недоліки алгоритму Шаркара .

15. Види планування та їх особливості.

Білет 2

1. Математична постановка задачі динамічного планування для багатопроцесорних систем.
2. Особливості операційних систем локальних обчислювальних мереж.
3. Що таке резидентні програми ОС?
4. Дати визначення динамічного планування.
5. Види модулів. Участь системних програм в перетворенні модулів.
6. Ідея алгоритму Янга.
7. Дати визначення програми оверлейної структури. Переваги, недоліки.
8. Вхідна й вихідна інформація редактору зв'язків.
9. Що таке неоднорідна система? Якими характеристиками визначається неоднорідність?
10. Перерахувати стратегії, що застосовуються в алгоритмах оптимізації.
11. Чим визначається перехід від непрямого методу доступу до колективного?.
12. Пояснити непряме введення-виведення.
13. Що таке супервизор й де він знаходиться?
14. Недолік генетичних алгоритмів.
15. Рівні планування в одно процесорній системі.

Білет 3

1. Недоліки методу колективного доступу.
2. Особливості розподілених операційних систем.
3. Структура модуля.
4. Стратегії, що використовуються при динамічному плануванні.
5. Що таке транзитні програми ОС?
6. Резиденція системи. Які програми там знаходяться?
7. Програма динамічної послідовної структури. Чим вона краще інших структур.
8. Коли застосовують евристичні алгоритми?
9. Дати визначення балансового планування. Які проблеми в його вирішенні?
10. Що таке резидентні вершини? Як їх знайти та як можна їх використати?
11. Недоліки використання мультипрограмування.
12. Основна відмінність використання підпрограм в компілюючих системах и інтерпретуючих.
13. Як знайти максимально й мінімально необхідну кількість процесорів?
14. Ідея застосування алгоритмів кластеризації. Шляхи визначення вузлів, що кластерізуються.
15. Навіщо потрібен рівень реконфігурації в системі планування багатопроцесорної паралельної системи?

Білет 4

1. Якими міркуваннями визначається вибір кванта при паралельній обробці?
2. Коли застосовують евристичні алгоритми?
3. Дати визначення області збереження. Чим визначається розмір, яка інформація завжди має бути збережена?
4. Особливості програмного забезпечення розподілених систем обробки інформації.
5. Відмінність завантажувального модуля від абсолютного. Які системні програми приймають участь у їх створенні?
6. Дати визначення ініціалізації ядра. Чим вона відрізняється від ініціалізації системи?
7. Чим визначається вибір змішаної дисципліни обслуговування?
8. Що називається ядром супервизора? Його функції.
9. Як визначити мінімально необхідну кількість процесорів для завантаження паралельного алгоритму? Дати приклад, коли формула дає неправильних результат.
10. Завантаження операційної системи. Етапи.?
11. Структура бібліотеки . Чим ця структура відрізняється від структури файлової системи?
12. Що таке віртуальна операційна система? Чи використовується вона в персональних комп'ютерах?
13. Основна характеристика задач планування з передуманням.
14. Перерахувати стратегії, що використовуються в алгоритмах оптимізації.
15. Алгоритми динамічного планування, що використовуються в сучасних ОС.

Білет 5

1. Дати визначення класичного мультипрограмування.
2. Дати визначення процесу. Стан процесу. Чим визначається вихід з активного в підготовлений стан?
3. Яка програма виконує запис інформації в область зберігання та яка інформація туди записується?
4. Чим відрізняється загрузка EXE й COM модулів?
5. Функції завантажника. В яких завантажниках які функції виконуються системою?
6. Чим відрізняються файлова система та бібліотека?
7. Як визначити об'єм пам'яті, необхідної для завантаження програми простої та оверлейної структури?
8. Чому алгоритм FBW краще RR? Яка характеристика при цьому покращується?
9. Дати визначення балансового планування. Які проблеми мають вирішуватися?
10. Що таке напівнеоднорідна система? Як виражається відношення «задача – ресурс»?
11. Що таке супервізор й де він знаходиться?
12. Дати визначення розподіленої операційної системи.
13. Які стратегії використовуються при розв'язанні NP-повних задач?
14. Дати визначення пошуку максимального паросполучення.
15. У чому відмінність часової складності від обчислювальної?

Білет 6

1. Дати визначення програми з точки зору операційної системи.
2. Умова переходу процесу з активного стану в підготований.
3. Зв'язок модулів за керуванням. Які операції виконуються й якими програмами? Яка інформація й куди при цьому записується?
4. Умови переходу з P3 у P1 або P2.
5. При наявності в'їзла, яке ребро береться у рішення?
6. Основні дії програми початкового завантаження. Де вона зберігається та як її знайти?
7. Як визначити об'єм пам'яті, необхідної для завантаження програми оверлейної структури?
8. Дати визначення ГРІД системи.
9. Як визначити зону знаходження оптимального розкладу для систем зі спільною пам'яттю?
10. Що таке строго неоднорідна ОС? Яка задача має розв'язуватися при плануванні?
11. Що шукає алгоритм Мальгранжа та який його основний недолік?
12. Яку задачу розв'язує лінійний планувальник? Яка стратегія при цьому використовується?
13. Чим відрізняється матриця зв'язності від матриці інцидентності?
14. Дати приклад неявної транзитності.
15. Засоби оцінки складності алгоритму.

Білет 7

1. Дати визначення непрямого доступу. Переваги й недоліки. Де він застосовується зараз.
2. Чому спулінг схильний до тупикових ситуацій?
3. Сформулювати принцип оптимальності Белмана?
4. Задачі, що розв'язуються при статичному плануванні.
5. Функції завантажника, що налаштовує. Яка програма готує для нього інформацію?
6. Де знаходиться програма початкового завантаження? Її функції.
7. Як визначити об'єм пам'яті, необхідної для завантаження програми динамічної послідовної структури?
8. Основна особливість пріоритетних дисциплін обслуговування. Види пріоритетів.
9. Що таке конфліктне значення? Навіщо їх виділяти?
10. Ідея оптимізації «базового» рішення.
11. Ідея алгоритмів покрокового конструювання.
12. Що таке NP-повна задача?
13. Основа алгоритму кластеризації, з застосуванням стратегії пошуку критичного шляху.
14. Способи опису графу.
15. Види топології паралельних систем. В якій топології можна досягнути мінімізації пересилок?

Білет 8

1. Дати визначення колективного доступу. Переваги й недоліки. Чи застосовується він зараз й де саме?
2. Умова переходу процесу з активного стану в заблокований.
3. Чому усі алгоритми, засновані на теоремі Бержа про максимальне паросполучення непридатні для використання в динамічних планувальниках?
4. Характеристика розподіленої операційної системи.
5. Дати визначення транзитного модуля. Чому вони, як правило, є СОМ-файлами?
6. Дати визначення табличного методу керування. Назвати першу таблицю, що заповнюється.
7. Цільова функція планування в реальному часі.
8. Яка характеристика покращується у змішаних алгоритмах обслуговування й чому її треба покращувати?
9. У чому відмінність між максимальним паросполученням й досконалим?
10. Що таке критичні вершини? Навіщо необхідно їх визначати та як вони визначаються?
11. Чим визначається зміст програм, що знаходяться в оперативній пам'яті?
12. У чому складність для операційної системи в організації багатoprogramного режиму?
13. Як визначити верхню границю кількості процесорів для завантаження комплексу задач?
14. Класифікація задач планування.
15. Як визначити мінімальну кількість процесорів?

Білет 9

1. Які основні характеристики розподілених систем обробки інформації?
2. Умова переходу процесу з заблокованого стану у готове.
3. Коли застосовують евристичні алгоритми?
4. Методи підвищення ефективності використання ресурсів обчислювальної системи.
5. Дати приклад «неявної» транзитності.
6. Що називається ядром супервізора, які його функції?
7. Яка системна програма готує інформацію для роботи завантажника?
8. Як визначити зону знаходження оптимального розкладу в системі із загальною шиною?
9. Навіщо потрібно виконувати структурний аналіз вихідного графа? Дати приклад.
10. Як змінюється вихідна інформація системи динамічного планування для однорідних і неоднорідних ОС?
11. У чому відмінність між максимальним паросполученням й досконалим?
12. Що таке пряме введення-виведення? Який метод застосовується в сучасних системах?
13. Дати визначення "спулінг". Навіщо він застосовується? У чому небезпека його застосування при введенні / виведенні?
14. Закон Амдала.
15. Критерії оптимізації рішення задач статичного планування.

Білет 10

1. Який механізм являється основою багатoprogramного режиму роботи?
2. Умова переходу процесу з активного стану у готовий або підготовлений.
3. Властивості модуля. Пояснити необхідність кожного.
4. Причини виникнення «безкінечного» відкладання. Чому воно небезпечне?
5. Дати визначення транзитного модуля. За якою схемою завантажуються транзитні модулі і в яку область пам'яті?
6. Які програми знаходяться у ядрі операційної системи? (Види програм)
7. Яку інформацію та як компілятор передає завантажнику, що налаштовує?
8. Чому NP-повні задачі не придатні для динамічного планування?
9. Чому й коли блокується система переривань?
10. Які методи використовуються для розв'язання NP-повних задач?
11. Що таке «спулінг», чому він схильний до тупикових ситуацій?
12. Дати визначення розподіленої системи.
13. Що таке базове рішення та у чому ідея його оптимізації?
14. Навіщо потрібно виділяти обов'язкові призначення?
15. Як визначити граничну кількість процесорів для завантаження програм у системі з загальною пам'яттю?

Білет 11

1. Дати визначення системного програмного забезпечення.

2. Особливості статичних планувальників.
3. Структура модуля.
4. У чому складність розв'язання задач планування паралельних процесів?
5. Що таке часова складність?
6. Причина переходу процесу з готового стану в активний.
7. Як визначити об'єм пам'яті, необхідної для завантаження програми динамічної послідовної структури?
8. Яка характеристика покращується у змішаних алгоритмах обслуговування?
9. Ідея алгоритмів, заснованих на стратегії критичного шляху.
10. Яку задачу вирішує потоковий планувальник? Коли й чому він неефективний?
11. Що таке перегляд команд вперед та яка характеристика покращується?
12. Пояснити непряме введення-виведення.
13. Які стратегії покращують роботу просторового планувальника?
14. Ідея створення адаптивних алгоритмів планування.
15. Як й коли визначається пріоритет системних та проблемних задач.

Білет 12

1. Перерахувати властивості модуля.
2. Особливості розподілених операційних систем. Чим вони відрізняються від ОС ЛВС?
3. Основні недоліки застосування мультипрограмування.
4. Методи організації обчислювального процесу, що підвищують продуктивність обчислювальної системи.
5. У чому смисл глобальної та поточної настройки адресних констант.
6. Закон Гроша.
7. Як визначити об'єм пам'яті, необхідний для завантаження програми простої структури?
8. Чому алгоритм «Корбата» краще FBn?
9. Де знаходяться нові PSW?
10. Ідея застосування методу оціночних функцій. Переваги й недоліки.
11. Недолік колективного доступу та як він усувається у сучасних системах?
12. Чому в розподілених системах переважніше використовувати крупнозернисте планування?
13. Пояснити термін «дружня операційна система».
14. Функції рівня операційної системи у багаторівневій системі програмування.
15. Описати рівні статичного планування для одно процесорної ОС.

Білет 13

1. Який тип транслятора використовується при трансляції з традиційної машинної мови у команди машини та як він реалізується?
2. Визначення метаобчислень (суперобчислень)
3. Чому в розподілених системах переважніше використовувати крупнозернисте планування?
4. Дії, що виконує програма ініціалізації ядра. Звідки вона викликається?
5. Функції завантажника, що налаштовує. Яку інформацію повідомляє йому компілятор?
6. Сформулювати теорему про обов'язкові призначення.
7. Як визначити об'єм пам'яті, необхідний для завантаження програми оверлейної структури?
8. Основна особливість пріоритетних дисциплін обслуговування. Види.
9. Дати визначення паралельної системи.
10. Ідея розв'язання задачі планування методом «пізнього» планування.
11. Задача «наповнення рюкзака». Для розв'язання якої задачі планування можна її застосовувати?
12. Яка інформація міститься у MBR?
13. Дати визначення процесу. Коли це поняття стало застосовуватися й чому?
14. Застосування GRID систем.
15. Відмінність завдання від задачі. Де зберігається завдання?

Білет 14

1. Що таке розв'язання задачі планування з передумовами?
2. Сформулювати теорему про обов'язкові призначення.
3. Яка програма виконує запис інформації в область збереження та яка інформація туди записується?
4. Обробка станів P3 й P4.

5. Який тип завантажника працює з модулями з розширеннями COM та EXE?
6. На яку характеристику впливають пріоритетні дисципліни обслуговування заявок?
7. Як визначити об'єм пам'яті, необхідний для завантаження програми динамічної паралельної структури?
8. Математична постановка задачі статичного планування.
9. Чому й коли блокується система переривань?
10. Як змінюється організація обчислювального процесу для систем зі спільною пам'яттю та SMP?
11. Сформулювати теорему про в'яло.
12. Що таке віртуальна операційна система, чи застосовується вона у сучасних ОС?
13. Проблеми керування паралельними процесами. Чому вони важкі для реалізації?
14. Для чого слугує характеристика «часова складність алгоритму»?
15. Що таке «строго неоднорідна ОС»? Дати приклад такої неоднорідності.

Білет 15

1. Дати характеристику неоднорідної системи.
2. Процес. Коли та як визначаються пріоритети системних й проблемних процесів?
3. Як передаються дані між модулями в оверлейних системах?
4. Які операції виконує завантажник при завантаженні EXE файлу?
5. Дати визначення резидентного модуля. Коли визначається склад резидентних програм?
6. Дати визначення табличного методу керування. Чому він застосовується?
7. Яка системна програма підготовлює інформацію для роботи завантажника, та яку?
8. Як визначити зону знаходження оптимального розкладу? Зі спільною пам'яттю та зі спільною шиною.
9. Структура супервізора.
10. Що таке транзитні програми? Спосіб їх завантаження.
11. Що таке запис у бібліотеку без каталогізації? Дати приклад.
12. Як змінюється інформація системи динамічного планування для однорідних та неоднорідних ОС?
13. Математична постановка задачі планування для одно процесорних систем.
14. Що таке розв'язання задачі планування з передумовами?
15. Принципи підвищення ефективності ОС.

Білет 16

1. Які міркування використовуються при визначенні величини кванта?
2. Умова переходу процесу з підготованого стану у готовий та з активного в підготований?
3. Чим відрізняється традиційна машинна мова програмування від команд машини?
4. Сенс розв'язання задачі реконфігурації при балансовому плануванні.
5. Що таке масштабування розподілених систем?
6. Чому застосовуються багаторівневі системи програмування?
7. Яку інформацію та як компілятор передає завантажнику, що налаштовує.
8. Функції редактора зв'язків. Як йому передається інформація?
9. Як визначити вершини графа, що знаходяться на критичному шляху?
10. Які методи використовуються для розв'язання NP-повних задач?
11. У чому відмінність розв'язання задачі планування від розподілення? Яка з них розв'язується при динамічному плануванні?
12. Функція мети задачі динамічного планування.
13. Сформулювати теорему про потужність паросполучення.
14. За якою схемою завантажуються транзитні програми супервізора?
15. Що таке межа Бременмана?

Білет 17

1. Дати визначення непрямого доступу. Чи застосовується він зараз й коли?
2. Що таке модульний принцип програмування?
3. Сформулювати принцип оптимальності Белмана.
4. Характеристика розподіленої операційної системи. Чим вона відрізняється від мережної?
5. Види завантажників та їх особливості. Коли який застосовується?
6. Які програми знаходяться у ядрі операційної системи? (Види програм)
7. Дати визначення програми простої структури. Коли використання такої структури неефективно?

8. Вхідна та вихідна інформація редактору зв'язків.
9. Як визначити вершини графу, що знаходяться на критичному шляху?
10. Основна ідея «угорського алгоритму». Заміна якого блоку покращує його характеристики?
11. Навіщо потрібно водити характеристику «надійний стан системи»?
12. Ідея алгоритмів, заснованих на стратегії критичного шляху.
13. Методи підвищення ефективності організації обчислювального процесу в ОС.
14. Що таке шлях, що збільшує, шлях, що чергується?
15. Які задачі розв'язуються при статичному плануванні у паралельній ОС?

Білет 18

1. Сформулювати принцип оптимальності Белмана.
2. Чи використовується у сучасних системах колективний доступ та чим він відрізняється від того, що використовувався раніше?
3. Умови створення процесу. Який ресурс при цьому є критичним?
4. Умови переходу зі стану P1 у P2 та назад.
5. Основні недоліки застосування мультипрограмування.
6. Чому у розподілених системах переважніше використовувати крупнозернисте планування?
7. Дати визначення програми оверлейної структури. Як передаються дані?
8. Сформулювати теорему про конфліктні призначення.
9. Дати визначення розподіленої операційної системи. Її відмінність від мережної.
10. Ідея створення адаптивних алгоритмів планування.
11. Особливості різних форм взаємодії «людина – машина». Які характеристики при цьому змінюються?
12. У чому складність для операційної системи в організації багатопрограмного режиму роботи? Які задачі про цьому розв'язуються та які механізми використовуються?
13. В якому вигляді компілятор передає інформацію завантажнику, що налаштовує?
14. Перерахувати рівні планування у паралельній системі. Які задачі потрібно розв'язувати у статичній, а які – у динамічній?
15. Що таке «строга неоднорідна ОС»? Які параметри ОС визначають неоднорідність?

Білет 19 [n/a]

1. Перерахувати способи реалізації багатопрограмного режиму роботи.
2. Сформулювати теорему про конфліктні призначення.
3. Вимоги до статичних планувальників.
4. Чим визначається пріоритет проблемних й системних програм та коли ці пріоритети визначаються?
5. Перерахувати ресурси ОС.
6. Дати визначення резидентного тому. Яка системна програма формує зміст резидентного тома?
7. Дати визначення програми динамічної послідовної структури.
8. Різниця між обчислювальною та часовою складністю. Яка оцінка коли застосовується?
9. Функція мети задачі статичного планування. Чому потрібно розділяти планування та розподілення?
10. Динамічне планування для систем реального часу.
11. Дати визначення керуючих програм операційної системи.
12. Що таке пряме введення-виведення? Чому непряме краще?
13. Як компілятор передає інформацію безпосередньо завантажнику, що зв'язує?
14. Обчислювальні системи реального часу. Визначення. Особливості
15. Назвати нижні рівні керування зовнішніми пристроями.

Білет 20 [n/a]

1. Чому для оцінки придатності алгоритму використовують часову складність, а не обчислювальну? 2. Чому усі алгоритми, засновані на теоремі Бержа про максимальне паросполучення непридатні для використання в динамічних планувальниках?
2. Що таке супервізор та де він знаходиться?
3. У чому відмінність визначення пріоритетів проблемних та системних програм?
4. Функції абсолютного завантажника.
5. Дати визначення резиденції системи. Чим визначається склад програм?
6. Дати визначення програми динамічної паралельної структури.

7. Коли застосовують евристичні алгоритми?
8. Як визначити конфліктне призначення?
9. Дати визначення розподіленої операційної системи.
10. Що таке «спулінг», коли він ефективний?
11. Ідея застосування алгоритмів кластеризації.
12. Яку задачу розв'язує просторовий планувальник?
13. У чому різниця завантаження СОМ та ЕХЕ програм?
14. Як визначити часову складність алгоритму?

Білет 21

1. Сформулювати принцип оптимальності Белмана.
2. Коли застосовують евристичні алгоритми?
3. Яка програма виконує запис інформації в область збереження та що там зберігається?
4. Що таке резидентні вершини та як їх знайти? Навіщо вони потрібні?
5. Функції завантажника, що налаштовує.
6. Дати визначення ініціалізації системи. Функції.
7. У чому відмінність меж максимальним паросполученням та досконалим?
8. Чому алгоритм «Корбато» краще FВп?
9. Що таке неоднорідна система? Як визначити степінь неоднорідності?
10. Функція мети задачі динамічного планування.
11. Сформулювати теорему про обов'язкові призначення.
12. Що таке «свопінг»?
13. Різниця між плануванням та розподіленням?
14. По якій характеристиці можна судити про придатність розробленого алгоритму для роботи у динамічному режимі?
15. Етапи завантаження операційної системи.

Білет 22

1. Дати визначення режиму розподілення часу.
2. Особливості розподілених операційних систем.
3. Зв'язок модулів по керуванню. Які операції виконуються та якими програмами?
4. Принципи підвищення ефективності роботи системи з допомогою організації обчислювального процесу.
5. Який тип завантажника працює з модулями з розширенням СОМ та ЕХЕ?
6. Різниця між плануванням й розподіленням?
7. Як визначити об'єм пам'яті, необхідний для завантаження програми оверлейної структури?
8. Основна особливість пріоритетних дисциплін обслуговування. Їх види.
9. Ідея оптимізації «базового» рішення.
10. Що таке конфліктне призначення? Навіщо потрібно його виділяти?
11. Що таке тимчасова бібліотека? Коли вона використовується?
12. Які операції виконує програма першого рівня планування?
13. Назвати рівні багаторівневих систем програмування.
14. Призначення операційних систем.
15. Задачі планування в системах масового розпаралелювання та чим вони відрізняються від задач планування для розподілених систем?

Білет 23

1. Умова переходу зі стану Р1 у Р2 й навпаки.
2. Які операції виконує програма першого рівня планування?
3. В чому перевага алгоритмів покрокового конструювання?
4. Чому для оцінки придатності алгоритму використовують тимчасову складність, а не обчислювальну?
5. Як визначити об'єм пам'яті, необхідної для завантаження програми динамічної-паралельної структури?
6. Назвати стратегії, що застосовуються в алгоритмах оптимізації.
7. Чому і коли блокується система переривань?
8. Основна ідея «угорського алгоритму».
9. Як змінюється вихідна інформація системи динамічного планування для однорідних і неоднорідних ОС?

10. Ідея розв'язання задачі планування методом «раннього» планування.
11. Що називається ядром супервізора, його функції?
12. Що таке віртуальна операційна система. Коли вона застосовується?
13. Дати визначення термінам «завдання» й «програма», дані з точки зору ОС.
14. Що таке квазі-оптимальне рішення?
15. Як підвищити ефективність роботи ОС зі спільною пам'яттю?

Білет 24 [п/а]

1. Дати визначення паралельної обробки. Недоліки та як їх позбутися?
2. Дати визначення процесу. Стратегія визначення пріоритетів в ОС.
3. Зв'язок модулів за керуванням. Які операції виконуються й якими програмами?
4. Особливості розв'язання задач в ОС ЛВС.
5. Види модулів. Які програми беруть участь в перетворенні модулів?
6. Дати визначення табличного методу керування. Як формується таблиця векторів переривань?
7. Як визначити об'єм пам'яті, необхідної для завантаження програми динамічної паралельної структури?
8. Чому алгоритм «Корбато» краще FВп?
9. Що таке критичні вершини?
10. Які операції виконує програма другого рівня планування?
11. Як змінюється організація обчислювального процесу для SMP і PC?
12. Ідея оптимізації «базового» рішення.
13. Яка характеристика покращується в змішаних алгоритмах обслуговування?
14. Дати визначення NP - повним завданням.
15. Різниця між плануванням і розподілом.

Білет 25

1. Дати визначення роботи в реальному часі. Надати приклад.
2. Різниця між плануванням і розподілом.
3. Чим відрізняється традиційна машинна мова програмування від команд машини?
4. У чому полягає сенс перегляду команд вперед і коли цей прийом неефективний?
5. Відмінність завантажувального модуля від абсолютного. Яка програма виконує перетворення?
6. Які операції виконує програма другого рівня планування?
7. Дати визначення програми оверлейної структури. Недоліки.
8. Вхідна і вихідна інформація редактора зв'язків.
9. Динамічне планування для систем реального часу.
10. Як визначити обов'язкове призначення? Навіщо це потрібно робити?
11. Що таке «розширення» розподілених систем?
12. Що таке «спонінг», чим відрізняється від «спулінга»?
13. Основні проблеми розв'язування задач планування в багатопроцесорних паралельних системах.
14. Описати послідовність дій, які виконуються завантажувачем при завантаженні програм.
15. Чому СОМ файли мають обмеження за розміром?

Білет 26

1. На що впливає введення поняття «надійний стан системи»?
2. Пропозиції Лебедева по підвищенню ефективності роботи ОС та як вони реалізовані в сучасних ОС.
3. Відмінність функцій програм системного введення /виведення від режиму спулінга.
4. Яка характеристика ОС покращується в змішаних дисциплінах обслуговування?
5. Частиною якої програми є редактор зв'язку?
6. Що таке супервізор й де він знаходиться?.
7. Навіщо потрібно виконувати структурний аналіз вихідного графа?
8. Функції 3-го рівня багаторівневих систем програмування.
9. Назвати рівні планування для багатопроцесорних ОС. Функції кожного.
10. Коли застосовується запис до бібліотеки без каталогізації?
11. Як визначити пріоритет процесу системного і проблемного?
12. Як змінюється організація обчислювального процесу для SMP і PC?
13. Функція мети завдання динамічного планування.

14. Ідея побудови «базового» рішення.
15. Сформулюйте принцип оптимальності Белмана.

Білет 27

1. Основні принципи організації обчислювального процесу, що підвищують ефективність роботи ОС.
2. Дати визначення режиму поділу часу.
3. Назвати базові рівні в багаторівневих системах програмування.
4. Визначити умови переходу зі стану P3 в стан P й P2.
5. Перерахувати пріоритетні дисципліни обслуговування. Яка характеристика впливає на вибір дисципліни?
6. Види завантажувачів та їх основні відмінності.
7. Чому спулінг схильний до тупиків?
8. Чому всі алгоритми, засновані на теоремі Бержа про максимальне паросполучення, не придатні для використання в динамічних планувальниках?
9. Особливості розподілених операційних систем. У чому їх відмінність від мережних?
10. Види модулів. Участь системних програм в перетворенні модулів.
11. Що таке обчислювальна складність?
12. Основна ідея «угорського алгоритму».
13. Чому не можна зациклити стан P3 й P4?
14. Коли застосовують евристичні алгоритми?
15. Яким чином і коли фіксується пріоритет процесів?

Білет 28

1. Чому впровадження логіки переривань зумовило підвищення продуктивності?
2. Що таке максимальне паросполучення?
3. Недолік непрямого методу доступу. Де і як зараз він використовується?
4. Дати визначення завдання, програми, даних.
5. Дати визначення ресурсу в сучасній ОС.
6. Перерахувати стратегії, що застосовуються в алгоритмах оптимізації.
7. Чим алгоритм FВп гірше алгоритму «Корбато»?
8. Сформулюйте принцип оптимальності Белмана.
9. Назвати стратегії вирішення задачі розподілу завдань по ресурсах з пересилками.
10. Як визначити зону знаходження оптимального розкладу з урахуванням пересилань?
11. Визначити умови переходу зі стану P2 в стан P1.
12. Навіщо потрібно виконувати структурний аналіз вихідного графа? Чи можна його робити в динамічному режимі?
13. Дати характеристику однорідної системи.
14. Математична постановка задачі динамічного планування.
15. Чому в розподілених системах переважніше використовувати крупнозернисте планування?

Відповіді

Білет 1

1. У чому сенс закону Гроша й чому він зараз не діє? #grosh

Сенс закону Гроша в тому, що продуктивність комп'ютера пропорційна квадрату його вартості. Зараз цей закон не діє, оскільки збільшення вартості комп'ютеру в 10 разів не означає збільшення продуктивності в 100 разів. Досягти потрібних потужностей складно.

2. На що впливає введення поняття «надійний стан системи»? #nadi

На запобігання тупиків та взаємного блокування процесів в системі.

3. Умови створення процесу. Хто створює процес. #proces

Умови створення процесу – коректність опису процесу на мові опису процесів та виділення ресурсів. Процес створює ОС.

4. Чому P3 и P4 не мають циклу? #p3 #p4 #cicl

Тому що P4 означає, що машина не працює, а P3 є атомарною операцією.

5. Види завантажників. Їх основні відмінності. #zavant

Абсолютні – тільки завантаження у пам'ять. Налаштовуючі – виділення пам'яті, налаштування зв'язування, завантаження у пам'ять. В кожному модулі потрібні внутрішні та зовнішні вектори. Безпосередньо зв'язуючі – тільки зв'язування.

6. Дати визначення ініціалізації системи. Мета ініціалізації #init #inic

Ініціалізація системи – ініціалізація установок та програм користувача.

7. Дати визначення програми простої структури. Переваги й недоліки. #pros #srtuct

Програма, яка при виконанні не звертається до інших програм, а має в своєму тілі все потрібне для виконання.

Перевага: швидкодія. Недоліки: займає більше місця, складніша для написання.

8. Функції редактора зв'язків. Звідки він бере інформацію для зв'язування? #red #zv

Редактор зв'язків збирає програму з багатьох об'єктних модулів та формує адреси у звертаннях до зовнішніх точок.

Назви модулів йому дає компілятор.

9. Дати визначення статичного планування. #static #stat #plan

План складається до вирішення задач та на іншому обладнанні. Важливою є якість, а не час.

10. Що таке критичний шлях та критичний час? #crit

Критичний шлях – максимальний шлях у графі. Критичний час – максимальний час проходження графу зверху вниз.

11. Що таке перегляд команд вперед та яка характеристика при цьому покращується? #command #comand

Попередня вибірка, покращується швидкодія.

12. Що таке 'свопінг'? #swap #swop

Свопінг – засіб реалізації багатопрограмного режиму роботи на одно процесорній машині. Проблема налаштування пов'язана з переміщенням програм в ОП. Якщо ОС працює зі свопінгом – потрібно налаштування адресних констант (глобальне – налаштування всіх адресних констант, локальне – обчислення адреси тієї змінної, яка знаходиться реально в ОП).

13. Перерахувати стратегії, що застосовуються в алгоритмах оптимізації. #optim #strat

Генетичних алгоритмів, оціночних функцій, прямого пошуку, Apealing метод, метод пошуку максимального паросполучення у зваженому графі, модифікований угорський, метод гілок та границь, модифікований метод для RealTimeOS, евристичний, складання розкладу, метод виключного планування.

14. Недоліки алгоритму Шаркара . #shark #algo

Необхідно проглядати весь граф, а це займає багато часу.

15. Види планування та їх особливості. #plan

* Статичне – час неважливий, план складається до вирішення задачі та на іншому обладнанні в інший час. *

Динамічне – проблема в швидкості, працює під час рішення задач на тому ж обладнанні в той же час. * Балансове – система планує навантаження, коли задача не вирішується – зовнішній планувальник намагається визначити чи правильне розподілення на вузлах. Проблеми – як визначити, що програма вісить? Передбачити звільнення, проблема міграції.

Білет 2

1. Математична постановка задачі динамічного планування для багатопроцесорних систем. #mat #math #plan

Задача комбінаторики по розміщенню задач в просторі та у площині.

2. Особливості операційних систем локальних обчислювальних мереж. #os #local #net

З'єднання незалежних пристроїв, високий рівень взаємозв'язку пристроїв, використання ЛМ для передачі інформації в цифровій формі.

3. Що таке резидентні програми ОС? #resident

Програми, що завжди знаходяться в ОП.

4. Дати визначення динамічного планування. #dynamic #plan

Динамічне планування – планування під час виконання задач на тому ж обладнанні і в той же час.

5. Види модулів. Участь системних програм в перетворенні модулів. #modul

Вихідний, об'єктний, завантажувальний, абсолютний. Компілятор між вихідним та об'єктним, редактор зв'язку між об'єктним та завантажувальним, завантажувач між завантажувальним та абсолютним.

6. Ідея алгоритму Янга. #yanga #janga #algo

Алгоритм Янга виділяє критичні вершини та піднімає їх (кластеризує).

7. Дати визначення програми оверлейної структури. Переваги, недоліки. #overlay

Задача, що розділяється на модулі. Переваги – простота написання, можливість використання модулів в різних програмах, модуль важить менше монолітної програми. Недоліки – потрібно описати взаємодію модулів, підключаючи їх, багато модулів займають більше місця.

8. Вхідна й вихідна інформація редактору зв'язків. #red #link

Вхідна – об'єктний модуль, вихідна – завантажувальний.

9. Що таке неоднорідна система? Якими характеристиками визначається неоднорідність? #system #neodnor

Заявка не може бути виконана на даному ресурсі.

10. Перерахувати стратегії, що застосовуються в алгоритмах оптимізації. #start #optim

Генетичних алгоритмів, оціночних функцій, напрямного пошуку, Anealing метод, метод пошуку максимального паросполучення у зваженому графі, модифікований угорський, метод гілок та границь, модифікований метод для RealTimeOS, евристичний, складання розкладу, метод виключного планування.

11. Чим визначається перехід від непрямого методу доступу до колективного?. #collect #dost #access

Тим, що реакція машини стає більш швидкою, ніж реакція користувача.

12. Пояснити непряме введення-виведення. #io #perp

Непряме введення-виведення – не сам процесор виконує ввід/вивід, а спеціальні процесори – канали. Це сталося після появи логіки переривань. Ввід/вивід розпаралелено з роботою основного процесору.

13. Що таке супервізор й де він знаходиться? #supervisor

Супервізор спостерігає за процесором від моменту створення до виходу. Він керує процесом, пам'яттю і роботою обладнання ОС. Знаходиться в ядрі.

14. Недолік генетичних алгоритмів. #genet #algo

Проблематично, коли необхідно знайти точний глобальний оптимум, час виконання функції великий, необхідно знайти всі рішення задач, конфігурація не є простою (кодування рішення).

15. Рівні планування в одно процесорній системі. #plan #level

Завдання проходять через три рівні. Перший – ті, що пройшли, претендують на захват ресурсів в системі. Середній рівень оброблює пріоритети і переводить завдання в ранг задач. Нижній рівень – визначення черги задач по виділенню часу процесора.

Білет 3

1. Недоліки методу колективного доступу. #collect

Смуга пропускання не належить одному користувачу, можуть бути проблеми з QoS.

2. Особливості розподілених операційних систем. #rozpod

Розподілена ОС – сукупність обчислювальних вузлів, з'єднаних між собою каналами зв'язку, з точки зору користувача представляє собою єдине ціле. Відсутність спільної пам'яті призводить до неможливості визначення загального стану за допомогою множини спільних змінних, а неможливість спільного звернення до пам'яті та різниця в затримках передач повідомлень призводить до того, що при визначенні стану будь-якого елементу системи з різних точок можна отримати різні результати. Виконання роботи розподіляється у вузлах, виходячи з міркувань пропускної здатності усієї системи. Розподіленні системи мають високий рівень організації паралельних обчислень.

3. Структура модуля Специфікація та тіло.

4. Стратегії, що використовуються при динамічному плануванні. #dynamic #strat

Пошук максимального паросполучення, визначення конфліктних призначень, виділення підматриць.

5. Що таке транзитні програми ОС? #transit

Які завантажуються у пам'ять тільки при необхідності.

6. Резиденція системи. Які програми там знаходяться? #resident

Це ОС, що отримується в результаті генерації. В ній знаходяться резидентні програми, тобто програми, що завжди в пам'яті.

7. Програма динамічної послідовної структури. Чим вона краще інших структур. #struct #posl

Програми, налаштовані за допомогою модульного принципу, представлені в переміщувальному вигляді, можуть підгружатися за мірою необхідності з організацією зв'язків по керуванню та даними. Краще інших, тому що пам'ять виділяється тільки коли потрібна.

8. Коли застосовують евристичні алгоритми? #evrist

Коли висока обчислювальна складність.

9. Дати визначення балансового планування. Які проблеми в його вирішенні? #balans #plan

Система планує навантаження, коли задача не вирішується – зовнішній планувальник намагається визначити чи правильне розподілення на вузлах. Проблеми – як визначити, що програма вісить? Передбачити звільнення, проблема міграції.

10. Що таке резидентні вершини? Як їх знайти та як можна їх використати? #resident

Вершини, що увійшли в критичний шлях, а їх часові та просторові координати не повинні змінюватись при вирішенні задач. Знайти критичний шлях.

11. Недоліки використання мультипрограмування. #multi

Може збільшитися час вирішення окремої задачі, завантаження процесорів збільшується, якщо задача захватить процесор, то вона його не віддає.

12. Основна відмінність використання підпрограм в компілюючих системах и інтерпретуючих. #program #compil #interp

Інтерпретуючі переводять частину програми в машинні команди і одразу виконують, виконуючи налаштування адресних констант. Компілючі створюють об'єктний модуль, який потім обробляється редактором зв'язків.

13. Як знайти максимально й мінімально необхідну кількість процесорів? #proc #min

Максимальна ширина ярусу ярусно-паралельної форми – максимальна кількість. Мінімальна: $N_{min} = \lceil \sum_{i=1}^n \{t_i\} / T_{kp} \rceil$

14. Ідея застосування алгоритмів кластеризації. #cluster #algo

Шляхи визначення вузлів, що кластеризуються. Зменшення графу. Знаходження критичної вершини, що визначає час вирішення.

15. Навіщо потрібен рівень реконфігурації в системі планування багатопроцесорної паралельної системи? #reconf #level

Для продовження роботи в разі виходу з ладу якогось обладнання.

Білет 4

1. Якими міркуваннями визначається вибір кванта при паралельній обробці? #kvant

Час кванту повинен бути більшим за час переключення процесів.

2. Коли застосовують евристичні алгоритми? #evrist

Коли висока обчислювальна складність задачі.

3. Дати визначення області збереження. Чим визначається розмір, яка інформація завжди має бути збережена? #obl #save #zberej

Це область пам'яті, в якій зберігаються регістри та відгалуження повернення. В неї записується інформація про перерваний процес і звідти система бере дані при відновленні процесу. Розмір залежить від кількості даних, які необхідні системі для відновлення процесу.

4. Особливості програмного забезпечення розподілених систем обробки інформації. #rozpod #distr

Відсутність спільної пам'яті призводить до неможливості визначення загального стану за допомогою множини спільних змінних, а неможливість спільного звернення до пам'яті та різниця в затримках передач повідомлень призводить до того, що при визначенні стану будь-якого елементу системи з різних точок можна отримати різні результати. Виконання роботи розподіляється у вузлах, виходячи з міркувань пропускну здатності усієї системи. Розподіленні системи мають високий рівень організації паралельних обчислень.

5. Відмінність завантажувального модуля від абсолютного. Які системні програми приймають участь у їх створенні? #modul #zavant #load #abs

Абсолютний модуль розміщується в пам'яті. Завантажувальний не може бути виконаним. Завантажувальний модуль отримується з об'єктного, оброблено редактором зв'язків. Абсолютний отримується з завантажувального, який оброблює завантажувач.

6. Дати визначення ініціалізації ядра. Чим вона відрізняється від ініціалізації системи? #kernel #init

Ініціалізація ядра – для формування системних таблиць, в яких зберігається вся інформація про програми супервізора та всіх параметрів, що динамічно змінюються. Ініціалізація системи – додавання додаткових функцій, які визначені користувачем.

7. Чим визначається вибір змішаної дисципліни обслуговування? #zmish #disc

Для покращення обслуговування. Використовуються різні комбінації щоб взяти найкращі сторони дисциплін.

8. Що називається ядром супервізора? Його функції. #supervis

Ядром супервізора є блок керування процесом. Він містить ім'я процесу, розподілені ресурси та контролює їх.

9. Як визначити мінімально необхідну кількість процесорів для завантаження паралельного алгоритму?

Дати приклад, коли формула дає неправильних результат. #min #proc

$$N_{min} = \lceil \sum_{i=1}^n \{t_i\} / T_{kp} \rceil$$

10. Завантаження операційної системи. Етапи.? #boot #os #load

Першим загрузається BIOS. Він тестує обладнання, завантажує таблиці переривань. Завантажується завантажувач MBR, який визначає кількість розділів, чи є на них ОС та яка. Потім загрузка передається ОС. Визначаються апаратні засоби, обирається конфігурація. Після цього відбувається загрузка ядра, його ініціалізація. Потім ініціалізація системи.

11. Структура бібліотеки . Чим ця структура відрізняється від структури файлової системи? #lib #fs #struct

Бібліотека, керуюча програма, каталог. Файлова система складається з каталогів та файлів.

12. Що таке віртуальна операційна система? Чи використовується вона в персональних комп'ютерах? #virt #os

Це ОС, яка може бути запущена з іншої ОС, при цьому їй виділяються або реальні апаратні засоби, або віртуальні. Це дозволяє на одній машині користуватися декількома користувачам різними ОС. Використовується.

13. Основна характеристика задач планування з передуманням. #plan #pered

Заздалегідь відомо що буде виконуватись (задача), на чому (ресурси), тому планування можна провести до виконання задач та на іншому обладнанні.

14. Перерахувати стратегії, що використовуються в алгоритмах оптимізації. #strat #optim

Генетичних алгоритмів, оціночних функцій, напрямного пошуку, Anealing метод, метод пошуку максимального паросполучення у зваженому графі, модифікований угорський, метод гілок та границь, модифікований метод для RealTimeOS, евристичний, складання розкладу, метод виключного планування.

15. Алгоритми динамічного планування, що використовуються в сучасних ОС. #dynam #plan

Алгоритм пошуку максимального паросполучення, виділення конфліктних призначень, пошук підматриць.

Білет 5

1. Дати визначення класичного мультипрограмування. #multiprog

Інаше именуется пакетным Задачи, планируемые к выполнению, называются пакетом. Переключение между задачами в пакетном режиме иницируется выполняющейся в данный момент задачей, поэтому промежутки времени выполнения той или иной задачи неопределены.

2. Дати визначення процесу. Стан процесу. Чим визначається вихід з активного в підготовлений стан?

#proc #state

Процессом является выполняемая программа, вместе с текущими значениями счетчика команд, регистров и переменных.



1. Процесс блокируется, ожидая входных данных
2. Планировщик выбирает другой процесс
3. Планировщик выбирает этот процесс
4. Доступны входные данные

Рис. 2.2. Процесс может находиться в состоянии выполнения, готовности или блокировки. Стрелками показаны возможные переходы между состояниями

3. Яка програма виконує запис інформації в область зберігання та яка інформація туди записується?

4. Чим відрізняється загрузка EXE й COM модулів? #com #exe #load

COM – абсолютный загрузчик, EXE – настраивающий. TLDR: COM не имеет хедеров и грузится с дефолтными настройками. EXE имеет кучу метаданных + таблицу линковки DLL + еще кучу ресурсов и тп. Там много математики вычисления сегментов. *Загрузка COM-программы* В процессе загрузки com-программы операционная система выполняет следующие действия: сегментные регистры CS, DS, ES, SS устанавливаются на начало PSP ; регистр SP устанавливается на конец сегмента PSP ; вся область памяти после PSP распределяется программе; в стек записывается слово 0000; указатель команд IP устанавливается на 100h (начало программы) с помощью команды JMP по адресу PSP :100h. *Загрузка EXE-программы* Загрузка exe-программы происходит значительно сложнее, так как связана с настройкой сегментных адресов: во внутренний буфер MS-DOS считывается форматированная часть заголовка файла; определяется размер загрузочного модуля определяется смещение начала загрузочного модуля в exe-файле вычисляется сегментный адрес для загрузки STARTSEG загрузочный модуль считывается в память по адресу STARTSEG:0000; сканируются элементы таблицы перемещений, располагающейся в exe-файле со смещением reltoff; для каждого элемента таблицы выполняются следующие действия: - считывается содержимое элемента таблицы как два двухбайтных слова (OFF, SEG); - вычисляется сегментный адрес ссылки перемещения - выбирается слово по адресу RELSEG:OFF, затем к этому слову прибавляется значение STARTSEG, после чего сумма записывается обратно по тому же адресу заказывается память для программы, исходя из значений minmem и maxmem; инициализируются регистры; программе передается управление. При инициализации регистры ES и DS устанавливаются на начало PSP , регистр AX устанавливается так же, как и для com-программ, в сегментный регистр стека SS записывается значение STARTSEG + ssreg, а в регистр SP записывается значение spreg. Для передачи управления программе в сегментный регистр CS записывается значение STARTSEG + csreg, а в регистр IP - значение ip_reg. Такая запись невозможна напрямую, поэтому операционная система сначала записывает в свой стек значение для CS, затем значение для IP и после этого выполняет команду дальнего возврата RETF (команда возврата из дальней процедуры).

5. Функції завантажника. В яких завантажниках які функції виконуються системою? #load #boot


Распределение памяти, настройка, редактирование, загрузка.

6. Чим відрізняються файлова система та бібліотека? #fs #lib

Бібліотека, керуюча програма, каталог. Файлова система складається з каталогів та файлів.

7. Як визначити об'єм пам'яті, необхідної для завантаження програми оверлейної структури? #struct

#memory #overlay

Теория: Оверлей - это такой способ использования оперативной памяти, при котором в один и тот же участок памяти, называемый оверлейным буфером, попеременно по мере надобности загружаются различные оверлейные (перекрывающиеся) модули. При этом все оверлейные модули в готовом к работе виде хранятся на диске, а в оперативной памяти в каждый момент находится лишь один активный модуль и, возможно, небольшое число неактивных. 

Ответ: Минимальный размер памяти = размер главного модуля + максимальный размер оверлейного модуля.

8. Чому алгоритм FBn краще RR? Яка характеристика при цьому покращується? #fbn #rr #algo

<http://www.lk.cs.ucla.edu/data/files/Kleinrock/Feedback%20Queueing%20Models%20for%20Time-Shared%20Systems.pdf> B

FBn существует N очередей. Заявка помещается в конец первой очереди. Процессор берет заявку из i-й очереди если очереди от 1 до i-1 пусты. После истечения кванта заявки идет в i+1 очередь Преимущество в том, что мелкие задачи быстрее выходят.

9. Дати визначення балансового планування. Які проблеми мають вирішуватися?

10. Що таке напівнеоднорідна система? Як виражається відношення «задача – ресурс»?

11. Що таке супервізор й де він знаходиться? #supervis

A supervisory program or supervisor is a computer program, usually part of an operating system, that controls the execution of other routines and regulates work scheduling, input/output operations, error actions, and similar functions and regulates the flow of work in a data processing system. It can also refer to a program that allocates computer component space and schedules computer events by task queuing and system interrupts. Control of the system is returned to the supervisory program frequently enough to ensure that demands on the system are met.

12. Дати визначення розподіленої операційної системи.

13. Які стратегії використовуються при розв'язанні NP-повних задач? #np #strat

Метод ветвей и границ состоит в отбрасывании заведомо неоптимальных решений целыми классами в соответствии с некоторой оценкой Метод локальных улучшений состоит в поиске более оптимального решения в окрестности некоторого текущего решения Приближенные и эвристические методы состоят в применении эвристик для выбора элементов решения Псевдополиномиальные алгоритмы представляют собой подкласс динамического программирования Метод случайного поиска состоит в представлении выбора последовательностью случайных выборов

14. Дати визначення пошуку максимального паросполучення. #max #parasp #search

Паросочетанием M называется набор попарно несмежных рёбер графа (иными словами, любой вершине графа должно быть инцидентно не более одного ребра из множества M). Мощностью паросочетания назовём количество рёбер в нём. Наибольшим (или максимальным) паросочетанием назовём паросочетание, мощность которого максимальна среди всех возможных паросочетаний в данном графе. Все те вершины, у которых есть смежное ребро из паросочетания (т.е. которые имеют степень ровно один в подграфе, образованном M), назовём насыщенными этим паросочетанием.

15. У чому відмінність часової складності від обчислювальної? #skladn #obig #bigo

Временная сложность – ф-я времени выполнения от размерности задачи. Вычислительная содержит 2 параметра – временную и пространственную (требование к памяти от размерности задачи)

Білет 6

1. Дати визначення програми з точки зору операційної системи. #program #os

Ничего адекватного пока не нашел

2. Умова переходу процесу з активного стану в підготований #state #active



1. Процесс блокируется, ожидая входных данных
2. Планировщик выбирает другой процесс
3. Планировщик выбирает этот процесс
4. Доступны входные данные

Рис. 2.2. Процесс может находиться в состоянии выполнения, готовности или блокировки. Стрелками показаны возможные переходы между состояниями

3. Зв'язок модулів за керуванням. Які операції виконуються й якими програмами? Яка інформація й куди при цьому записується?


4. Умови переходу з P3 у P1 або P2. #perehid #p3 #p1 #p2

Выход из прерывания, вызов сист. Функции из прерывания P4 - машина не работает, выхода из него нет P3 -

состояние, в которое переходит если зафиксировано прерывание. атомарная операция расшифровки прерывания P2 - программы пользователя (прикладные) P1 - система обрабатывает системные процессы

5. При наявності віяла, яке ребро береться у рішення?

6. Основні дії програми початкового завантаження. Де вона зберігається та як її знайти? #boot #load POST(проверка оборудования). Загрузка бутового загрузчика и передача ему управления. Лежит в BIOS.

7. Як визначити об'єм пам'яті, необхідної для завантаження програми оверлейної структури? #struct #overlay
Теория: Оверлей - это такой способ использования оперативной памяти, при котором в один и тот же участок памяти, называемый оверлейным буфером, попеременно по мере надобности загружаются различные оверлейные (перекрывающиеся) модули. При этом все оверлейные модули в готовом к работе виде хранятся на диске, а в оперативной памяти в каждый момент находится лишь один активный модуль и, возможно, небольшое число неактивных.  Ответ: Размер памяти = размер главного модуля + максимальный размер оверлейного модуля. Если модули вызывают друг-друга, то максимальная сумма одновременно загруженных модулей

8. Дати визначення ГРІД системи.

Он точно это давал на лекции, нужно просмотреть конспекты

9. Як визначити зону знаходження оптимального розкладу для систем зі спільною пам'яттю?

10. Що таке строго неоднорідна ОС? Яка задача має розв'язуватися при плануванні? #neodnor #os
http://www.multicoreinfo.com/research/papers/whitepapers/OSHMA_2007.pdf Проблемы: 1. Optimal performance (Производительность) 2. Core assignment balance(Баланс нагрузки) 3. Response time fairness.

11. Що шукає алгоритм Мальгранжа та який його основний недолік? #malgrandj #nirvana #algo
разбиение графов на сильно связанные под-графы.

12. Яку задачу розв'язує лінійний планувальник? Яка стратегія при цьому використовується?

13. Чим відрізняється матриця зв'язності від матриці інцидентності? #matrix

Связность показывает связь между i-й и j-й вершинами Инцидентность показывает соседство вершин и ребер(строка – вершина, столбец – ребро)

14. Дати приклад неявної транзитності. #transit #image



Білет 7

1. Дати визначення непрямого доступу. Переваги й недоліки. Де він застосовується зараз. #acces #dost #nepr
Режим буферизации для выравнивания скоростей при вводе и считывании информации из буфера. При работе программ системного ввода и/режим спуллинга – согласование скоростей на входе и выходе.

2. Чому спулінг схильний до тупикових ситуацій? #spul #tupik #sssr

1. Спулинг - это выравнивание скоростей передачи данных внешних устройств за счет буферизации. 2. В алгоритмах кластеризации схожие по функциональным признакам узлы объединяются в группы - кластеры

3. Сформулювати принцип оптимальності Белмана?. #belman #optim #princ

Эвристические (основанные на опыте людей), Пошаговое конструирование – разделение всего процесса решения на определенное кол-во шагов и нахождение оптимального или квазиоптимального решения. ЭТО – соответствует принципу оптимальности Белмана.(???)

4. Задачі, що розв'язуються при статичному плануванні. #static #plan

Поиск min-го кол-ва процессоров, необходимых для решения комплекса информационно- и по управлению взаимосвязанных задач за время, не превышающее заданное или критическое; поиск плана решения заданного комплекса информационно- и по управлению взаимосвязанных задач на заданном кол-ве процессоров за минимальное время.

5. Функції завантажника, що налаштовує. Яка програма готує для нього інформацію? #loader #comp #func

Распределение памяти, настройка, редактирование, загрузка. Информацию для него готовит компилятор.

6. Де знаходиться програма початкового завантаження? Її функції. #loader #func #mbr
В MBR. Активизация программ начальной загрузки.

7. Як визначити об'єм пам'яті, необхідної для завантаження програми динамічної послідовної структури? #struct #memory #dynamic
Корневой сегмент + наибольший суммарный объем модулей, которые могут выполняться параллельно

8. Основна особливість пріоритетних дисциплін обслуговування. Види пріоритетів. #prior #disc
Заявки имеют приоритет. Приоритетное обслуживание – заявка на вход системы с заданным приоритетом. Относительный приоритет – не может прервать задачу на ресурсе, даже если она имеет низкий приоритет. Абсолютный приоритет – прерывает задачу на ресурсе если та имеет более низкий приоритет. Динамический приоритет – ЕСЛИ возникает опасность бесконечного откладывания. ЕСЛИ по мнению sys, заявка слишком долго занимает ресурс, ТО ее приоритет понижается. ЕСЛИ заявка очень долго ожидает ресурс ТО ее приоритет повышается.(QoS)

9. Що таке конфліктне значення? Навіщо їх виділяти? #conflict

1. Конфликтное назначение уменьшает мощность паросочетания. Это назначения стоящие под главной диагональю после выделения основных подматриц.

10. Ідея оптимізації «базового» рішення. #base #optimis

2. «Базовое» решение оптимизируется по необходимому критерию - время выполнения, количество процессоров.

11. Ідея алгоритмів покрокового конструювання.

12. Що таке NP-повна задача? #np

Та, яка не вирішується за поліноміальний час, по ходу

13. Основа алгоритму кластеризації, з застосуванням стратегії пошуку критичного шляху.(??) #clast #algo
Зменшення графу. Знаходження критичної вершини, що визначає час вирішення.

14. Способи опису графу. #graph

Табличный, графический, аналитический и словесный.

15. Види топології паралельних систем. В який топології можна досягнути мінімізації пересилок? #topol
#paral

PARA-PC Velsh – Рисунок – много ОЗУ много ПЭ – все подключены к облаку – Коммуникационная среда – (Статическая и динамическая коммутация(ПЭ-ПЭ или ПЭ ОЗУ)) => Можно моделировать (Общую Память) (Общая Шина) (КЭШ-шинно ориентированные – Общая шина + Каждый ПЭ общается с шиной via КЭШ (содержит программный код и промежуточные данные) (Система с виртуальной памятью – предыдущее без ОП))Дерево Гипергуб

Білет 8

1. Дати визначення колективного доступу. Переваги й недоліки. Чи застосовується він зараз й де саме?

#access #collectiv #colectiv

Коллективный доступ Предусматривающий доступ к ресурсам системы (система работает в многопрограммном режиме) многих пользователей, одновременная работа нескольких users на машине (Логика прерываний). Увеличивается скорость. Надежность машины. Доступ к ресурсам системы имеют несколько пользователей, система работает в многопрограммном режиме. У каждого пользователя свой терминал. Система должна быть очень надежной, а центральная машина - очень мощной.

2. Умова переходу процесу з активного стану в заблокований #perekid #state #proc

Истечение кванта времени. (Выделенное время процессора для выполнения этого процесса завершилось и процессор занят другим процессом).

3. Чому усі алгоритми, засновані на теоремі Бержа про максимальне паросполучення непридатні для використання в динамічних планувальниках #berj #berg

Пусть зафиксировано некоторое паросочетание М. Тогда простая цепь $P = (v_1, v_2, \dots, v_k)$ называется чередующейся цепью, если в ней рёбра по очереди принадлежат - не принадлежат паросочетанию М. Чередующаяся цепь называется увеличивающей, если её первая и последняя вершины не принадлежат паросочетанию. Иными словами, простая цепь Р является увеличивающей тогда и только тогда, когда вершина v_1

!in M, ребро (v_2, v_3) in M, ребро (v_4, v_5) in M, ..., ребро (v_{k-2}, v_{k-1}) in M, и вершина v_k !in M.



Теорема Бержа (Claude Berge, 1957 г.). Паросочетание M является наибольшим тогда и только тогда, когда для него не существует увеличивающей цепи.

Доказательство необходимости. Пусть для паросочетания M существует увеличивающая цепь P. Покажем, как перейти к паросочетанию большей мощности. Выполним чередование паросочетания M вдоль этой цепи P, т.е. включим в паросочетание рёбра (v_1, v_2) , (v_3, v_4) , ..., (v_{k-1}, v_k) , и удалим из паросочетания рёбра (v_2, v_3) , (v_4, v_5) , ..., (v_{k-2}, v_{k-1}) . В результате, очевидно, будет получено корректное паросочетание, мощность которого будет на единицу выше, чем у паросочетания M (т.к. мы добавили $k/2$ рёбер, а удалили $k/2 - 1$ ребро).

Доказательство достаточности. Пусть для паросочетания M не существует увеличивающей цепи, докажем, что оно является наибольшим. Пусть M — наибольшее паросочетание. Рассмотрим симметрическую разность $G = M (+) M$ (т.е. множество рёбер, принадлежащих либо M, либо M, но не обоим одновременно (WTF?!)).

Покажем, что G содержит одинаковое число рёбер из M и M (т.к. мы исключили из G только общие для них рёбра, то отсюда будет следовать и $|M| = |M|$). Заметим, что G состоит только из простых цепей и циклов (т.к. иначе одной вершине были бы инцидентны сразу два ребра какого-либо паросочетания, что невозможно). Далее, циклы не могут иметь нечётную длину (по той же самой причине). Цепь в G также не может иметь нечётную длину (иначе бы она являлась увеличивающей цепью для M, что противоречит условию, или для M, что противоречит его максимальности). Наконец, в чётных циклах и цепях чётной длины в G рёбра поочередно входят в M и M, что и означает, что в G входит одинаковое количество рёбер от M и M. Как уже упоминалось выше, отсюда следует, что $|M| = |M|$, т.е. M является наибольшим паросочетанием. Теорема Бержа даёт основу для алгоритма Эдмондса — поиск увеличивающих цепей и чередование вдоль них, пока увеличивающие цепи найдутся.

4. Розподілена система — це набір незалежних комп'ютерів, що представляється їх користувачам єдиною об'єднаною системою. #rozpod

У цьому визначенні обмовляються два моменти. Перший відноситься до апаратури: всі машини автономні. Другий стосується програмного забезпечення: користувачі думають, що мають справу з єдиною системою. Важливо обидва моменти. Можливо, замість того щоб розглядати визначення, розумніше буде зосередитися на важливих характеристиках розподілених систем. Перша з таких характеристик полягає в тому, що від користувачів приховані відмінності між комп'ютерами і способи зв'язку між ними. Те ж саме відноситься і до зовнішньої організації розподілених систем. Іншою важливою характеристикою розподілених систем є спосіб, за допомогою якого користувачі і додатки одночасно працюють в розподілених системах, незалежно від того, де і коли відбувається їх взаємодія.

Розподілені системи повинні також відносно легко піддаватися розширенню, або масштабуванню. Ця характеристика є прямим наслідком наявності незалежних комп'ютерів, але в той же час не указує, яким чином ці комп'ютери насправді об'єднуються в єдину систему. Розподілені системи зазвичай існують постійно, проте деякі їх частини можуть тимчасово виходити з ладу. Користувачі і додатки не повинні повідомлятися про те, що ці частини замінені або пошкоджені або що додані нові частини для підтримки додаткових користувачів або додатків.

Для того, щоб підтримати представлення різних комп'ютерів і мереж у вигляді єдиної системи, організація розподілених систем часто включає додатковий рівень програмного забезпечення, що знаходиться між верхнім рівнем, на якому знаходяться користувачі і додатки, і нижнім рівнем, що складається з операційних систем, як показано на рис. 1. Відповідно, така розподілена система зазвичай називається системою проміжного рівня (middleware).

5. Дать определение – транзитный модуль. #transit

Программы, связанные с выполнением функций ОС, но не находящиеся постоянно в ОП называются транзитными. Эти программы вызываются в ОП по мере необходимости. – проги., которые могут понадобиться 4/ выполнения ф-ций ОС (но не резидентные) – могут вызываться по оверлейной || динамически-последовательной схеме в транзитную зону.

6. Дать определение Табличный метод управления #table

Метод когда решение управляющей системы принимается на основании инфы, хранящейся в таблицах, содержащих данные состояния всех частей системы.

7. Цільова функція планування в реальному часі.

8. Яка характеристика покращується у змішаних алгоритмах обслуговування й чому її треба покращувати? #mixed #algo

Среднее время ожидания заявки

9. Найти максимальное паросочетание #para #max

- значит найти максимальное число ребер графа в которых не совпадают координаты вершин, либо найти максимальное число единиц матрицы у которых не совпадают координаты. С

10.Що таке критичні вершини?

11. . Чим визначається зміст програм, що знаходяться в оперативній пам'яті?

12. У чому складність для операційної системи в організації багатопрограмного режиму? #multi #os

Организация защиты от взаимного влияния друг на друга на уровне оперативной и на уровне внешней памяти; разделение аппаратных и программных ресурсов; планирование (во времени, а в случае ПВС и в пространстве).

13.Как определить верхнюю границу кол-ва процессоров для погружения комплекса задач . #max #proc

Можно определить max число процессоров, выше которого практически не выгодно иметь больше процессоров – max ширина яруса.

14.Классификация задач планирования(?) #plan

Динамическое Статическое Балансовое В реальном времени

15. минимальное к-во процессоров, #proc #min

когда время выполнения взаимосвязанных задач не превышает $T_{кр} N_{min} = [SUM(i=1, n) \{t_i\} / T_{кр}]$

Білет 9

1.Розподілена система #distr #sys

— це набір незалежних комп'ютерів, що представляється їх користувачам єдиною об'єднаною системою. У цьому визначенні обмовляються два моменти. Перший відноситься до апаратури: всі машини автономні. Другий стосується програмного забезпечення: користувачі думають, що мають справу з єдиною системою. Важливо обидва моменти. Можливо, замість того щоб розглядати визначення, розумніше буде зосередитися на важливих характеристиках розподілених систем. Перша з таких характеристик полягає в тому, що від користувачів приховані відмінності між комп'ютерами і способи зв'язку між ними. Те ж саме відноситься і до зовнішньої організації розподілених систем. Іншою важливою характеристикою розподілених систем є спосіб, за допомогою якого користувачі і додатки одночасно працюють в розподілених системах, незалежно від того, де і коли відбувається їх взаємодія.

Розподілені системи повинні також відносно легко піддаватися розширенню, або масштабуванню. Ця характеристика є прямим наслідком наявності незалежних комп'ютерів, але в той же час не указує, яким чином ці комп'ютери насправді об'єднуються в єдину систему. Розподілені системи зазвичай існують постійно, проте деякі їх частини можуть тимчасово виходити з ладу. Користувачі і додатки не повинні повідомлятися про те, що ці частини замінені або пошкоджені або що додані нові частини для підтримки додаткових користувачів або додатків.

Для того, щоб підтримати представлення різних комп'ютерів і мереж у вигляді єдиної системи, організація розподілених систем часто включає додатковий рівень програмного забезпечення, що знаходиться між верхнім рівнем, на якому знаходяться користувачі і додатки, і нижнім рівнем, що складається з операційних систем, як показано на рис. 1. Відповідно, така розподілена система зазвичай називається системою проміжного рівня (middleware).

2. Условие перехода процесса из блокированного состояния в готовое. #block #proc #perh #state

Наступление события, освобождения ресурса, завершение ввода/вывода.

3. В інформатиці, евристичний алгоритм, #everist

або просто евристика, це алгоритм спроможний видати прийнятне рішення проблеми серед багатьох рішень, але неспроможний гарантувати, що це рішення буде найкращим. Отже такі алгоритми є приблизними і неточними. Зазвичай такі алгоритми знаходять рішення близьке до найкращого і роблять це швидко. Іноді такий алгоритм може бути точним, тобто він знаходить дійсно найкраще рішення, але він все одно буде називатися евристичним, доти доки не буде доведено, що рішення дійсно найкраще. Один з найвідоміших жадібний алгоритм, для того, щоб бути простим і швидким цей алгоритм ігнорує деякі вимоги задачі.

Дві фундаментальні цілі в інформатиці - знаходження алгоритмів з вірогідно найкращим часом виконання та з хорошою або оптимальною якістю. Евристичний алгоритм відмовляється від однієї або обох цих цілей;

наприклад, він зазвичай знаходить дуже хороше рішення, але немає доказів, що рішення насправді не є поганим; або працює досить швидко, але не має гарантії, що він завжди видасть рішення. Декілька евристичних методів використовуються антивірусним ПЗ для виявлення вірусів та іншого шкідливого ПЗ. Часто, можна знайти таку задачу, що евристичний алгоритм буде працювати або дуже довго, або видавати невірні результати, однак, такі парадоксальні приклади можуть ніколи не зустрітись на практиці через свою специфічну структуру. Таким чином, використання евристики дуже поширене в реальному світі. Для багатьох практичних проблем евристичні алгоритми, можливо, єдиний шлях для отримання гарного рішення в прийнятний проміжок часу. Існує клас евристичних стратегій названих метаалгоритмами, котрі часто використовують, наприклад, випадковий пошук. Такі алгоритми можуть бути застосовані до широкого кола завдань, при цьому хороші характеристики не гарантуються.

4. Методи підвищення ефективності використання ресурсів обчислювальної системи.

5. Дати приклад «неявної» транзитності. #transit #image



6. Що називається ядром супервізора, які його функції? #supervis #kernel

Ядром супервізора є блок керування процесом. Він містить ім'я процесу, розподілені ресурси та контролює їх.

7. Яка системна програма готує інформацію для роботи завантажника #loader #compiler

Компілятор.

8. Як визначити зону знаходження оптимального розкладу в системі із загальною шиною?

9. Навіщо потрібно виконувати структурний аналіз вихідного графа? Дати приклад.

10. Як змінюється вихідна інформація системи динамічного планування для однорідних і неоднорідних ОС?

11. У чому відмінність між максимальним парасполученням й досконалим? #parasp #vidm

Найти максимальное паросочетание - значит найти максимальное число ребер графа в которых не совпадают координаты вершин, либо найти максимальное число единиц матрицы у которых не совпадают координаты.

12. Що таке пряме введення-виведення? Який метод застосовується в сучасних системах? #io

Когда не используются управляющие программы. Пользовательский процесс сам осуществляет i/o

13. Дати визначення "спулінг". Навіщо він застосовується? У чому небезпека його застосування при введенні / виведенні? #spul

Режим буферизации для выравнивания скоростей при вводе и считывании информации из буфера. При работе программ системного ввода и/или режим спуллинга – согласование скоростей на входе и выходе.

14. Закон Амдала. #amdala #amnedala

Закон Амдала (англ. Amdahl's law, иногда также Закон Амдаля-Уэра) — иллюстрирует ограничение роста производительности вычислительной системы с увеличением количества вычислителей. Джин Амдал сформулировал закон в 1967 году, обнаружив простое по существу, но непреодолимое по содержанию ограничение на рост производительности при распараллеливании вычислений: «В случае, когда задача разделяется на несколько частей, суммарное время её выполнения на параллельной системе не может быть меньше времени выполнения самого длинного фрагмента». Согласно этому закону, ускорение выполнения программы за счёт распараллеливания её инструкций на множестве вычислителей ограничено временем, необходимым для выполнения её последовательных инструкций.

Предположим, что необходимо решить некоторую вычислительную задачу. Предположим, что её алгоритм таков, что доля от общего объёма вычислений может быть получена только последовательными расчётами, а, соответственно, доля может быть распараллелена идеально (то есть время вычисления будет обратно пропорционально числу задействованных узлов). Тогда ускорение, которое может быть получено на вычислительной системе из процессоров, по сравнению с однопроцессорным решением не будет превышать величины

$$S_p = 1 / \{ \alpha + \{ 1 - \alpha \} / p \}$$

Закон Амдала показывает, что прирост эффективности вычислений зависит от алгоритма задачи и ограничен сверху для любой задачи с $\alpha \neq 0$. Не для всякой задачи имеет смысл наращивание числа процессоров в вычислительной системе.

Более того, если учесть время, необходимое для передачи данных между узлами вычислительной системы, то зависимость времени вычислений от числа узлов будет иметь максимум. Это накладывает ограничение на

масштабируемость вычислительной системы, то есть означает, что с определенного момента добавление новых узлов в систему будет увеличивать время расчёта задачи.

15. Критерії оптимізації рішення задач статичного планування #optim #crit #static #plan

Статическое планирование – план составляется на другой машине &&|| в другое время.

Оптимальный(оптимизированный) план за обозримое (физически нормальное время) Критерий оптимальности – время решения задачи и минимизация и/ресурсов. (NP-полная – временная сложность - экспонента)

Білет 10

1. Який механізм являється основою багатопрограмного режиму роботи? #multi

Механізм переривань

2. Умова переходу процесу з активного стану у готовий або підготовлений? #state #active #ready

Коли процес довго виконується та двигается достатньо далеко, то планувальник процесів вирішує, що процесор необхідно віддати іншому процесу, після чого переводить цей процес в стан підготовка та віддає процесор іншому. Коли ресурс виділяється процесу, то він переходить в стан готовий.

3. Властивості модуля. Пояснити необхідність кожного. #module

Властивості: стандартна внутрішня структура, взаємна незалежність модулів, функціональна завершеність, параметрична універсальність. Вихідний модуль містить в собі код модуля, який потім буде виконуватись. Об'єктний модуль містить в собі скомпільований код, який потім може бути об'єднаний з іншими модулями для виконавчого модуля або бібліотеки. Завантажувальний модуль готовий до виконання, але в ньому не налаштовані адресні константи, написаний на машино-орієнтованій мові. Абсолютний модуль є виконавчим, в ньому визначені адресні константи.

4. Причини виникнення «безкінечного» відкладання. Чому воно небезпечне? #unlim #infinite #danger #warning

Безкінечне відкладання може виникнути якщо процес має малий пріоритет коли планувальник розподіляє ресурси по пріоритетах. Постійно будуть приходити більш пріоритетні процеси, а цей буде безкінечно довго очікувати виділення ресурсів. Це небезпечно тим, що безкінечне очікування може призвести до тупика, який важко відловити.

5. Дати визначення транзитного модуля. За якою схемою завантажуються транзитні модулі і в яку область пам'яті? #transit

Транзитний модуль – модуль, який завантажуються в пам'ять тільки тоді, коли він потрібен. Завантажуються по оверлейній або динамічно-послідовній схемі в транзитну область. Але зараз завантажуються в будь-яку область ОП.

6. Які програми знаходяться у ядрі операційної системи? (Види програм) #kernel #prog

Програми, що оброблюють переривання, контролюють процеси, забезпечують їх взаємодію, підтримують операції вводу/виводу, підтримують роботу файлової системи, диспетчеризація процесів, задач, ресурсів, ведення обліку роботи машини.

7. Яку інформацію та як компілятор передає завантажнику, що налаштовує? #boot #loader #compiler

Компілятор передає інформацію про адресні константи в даній команді за допомогою бітів перемістимості.

8. Чому NP-повні задачі не придатні для динамічного планування? #np #dynamic #plan

Тому що вони не можуть бути вирішені за поліноміальний час n^k . А при динамічному плануванні основна ціль – зменшити час планування, тому динамічні алгоритми мають як правило лінійну складність.

9. Чому й коли блокується система переривань? #int #interrupt #bloc

При дешифрації поточного переривання, тому що дешифрація є атомарною операцією.

10. Які методи використовуються для розв'язання NP-повних задач? #np #method

Евристичні алгоритми, псевдополіноміальні, метод локальних покращень, метод гілок та границь, метод випадкового пошуку.

11. Що таке «спулінг», чому він схильний до тупикових ситуацій? #spul

Спулінг – на вході в систему обслуговування одна швидкість, на виході інша. Задачам при цьому створюється ілюзія одночасного доступу до пристроїв, але задачі працюють без прямого доступу до пристроїв. Спулінг допомагає уникнути тупиків, оскільки напряму з пристроєм працює лише демон.

12. Дати визначення розподіленій системі. #distr

Розподілена ОС – сукупність обчислювальних вузлів, з'єднаних між собою каналами зв'язку, з точки зору користувача представляє собою єдине ціле.

13. Що таке базове рішення та у чому ідея його оптимізації? #base #optimis

Базове рішення – перше правильне рішення задачі. Оптимізація його зменшує час вирішення задачі.

14. Навіщо потрібно виділяти обов'язкові призначення? #prizn #obov

Для того, щоб знайти максимальне паросполучення та щоб можна було виявити конфліктні призначення.

15. Як визначити граничну кількість процесорів для завантаження програм у системі з загальною пам'яттю? #proc #max #min

Максимальна ширина ярусу ярусно-паралельної форми – максимальна кількість. Мінімальна: $N_{min} = \lceil \sum_{i=1}^n t_i \rceil / T_{kp}$

Білет 11

1. Дати визначення системного програмного забезпечення. #sys

Системное программное обеспечение – совокупность программ и их описание для обеспечения функционирования вычислительной системы. Ф-ции: Обеспечить эффективное прохождение заданий usera через систему (с точки зрения usera – среднее время ожидания и производительность) и повышения эффективности.

2. Особливості статичних планувальників. #stat #plan

План решения задач может составляться на другом оборудовании и/или в другое время

3. Структура модуля

4. У чому складність розв'язання задач планування паралельних процесів? #plan #parallel #paralel

Добавляется планирование в пространстве; появляется проблема синхронизации процессов. ЕСЛИ в системе нет одинаковых устройств то очереди обрабатываются по линейной схеме => планирование только во времени ИНАЧЕ планирование во времени и в пространстве.

5. Временная сложность –зависимость времени решения задачи от размерности задачи. (Берем наихудший случай) #skl #time

6. Причина переходу процесу з готового стану в активний. #state #activ #ready

Выделение кванта времени процессора, запуск

7. Як визначити об'єм пам'яті, необхідної для завантаження програми динамічної послідовної структури? #dynamic #struct

Память выделяется по мере надобности – по цепочке. Уровни не выделяются. Корневой модуль всегда в памяти

8.

9.

10.

11.

12. Пояснити непряме введення-виведення #io

Используется единое обращение к памяти ВУ, процессор освобождается от управления ВУ, и функция ЦП заключается в инициализировании канальных программ и завершении организации ввода/вывода с помощью канала (многопроцессорная система).

13.

14. Ідея створення адаптивних алгоритмів планування #adaptiv #plan

Я думаю, (а значить, існую) динамическое назначение приоритетов

15. Як й коли визначається пріоритет системних та проблемних задач. #prior #sys

Приоритеты системных процессов – при инициализации системы. Блоки управления проблемными

(пользовательскими) процессами создаются в процессе активизации процессов динамически – по идее, в этот момент и назначаются приоритеты

Білет 12

1. Перерахувати властивості модуля. #modul

Стандартность внутренней структуры, функциональная завершенность, параметрическая универсальность, взаимная независимость

2. Особливості розподілених операційних систем. Чим вони відрізняються від ОС ЛВС? #dist #os #sys

Распределенная sys – совокупность выч. узлов, связанных между собой каналами связи, с точки зрения usera представляют собой единое целое. Отсутствие общей памяти приводит к невозможности определения общего состояния с помощью множества совместных переменных, а невозможность совместного обращения к памяти и различие в задержках передач сообщений приводит к тому что при определении состояния какого либо элемента системы из двух различных точек можно получить разные результаты. Выполнение работы распределяется в узлах исходя из соображения пропускной способности всей системы. Распределенные системы имеют высокий уровень организации параллельных вычислений

3. Основні недоліки застосування мультипрограмування.

4. Методи організації обчислювального процесу, що підвищують продуктивність обчислювальної системи. #prod #sys #org

Нашел только производительность файловой системы: -кеширование(использовать буфера в ОП) –для быстрого поиска в кеше используется хеш таблица * опережающее чтение блока * снижение времени перемещения блока головок * файловая система с журнальной структурой LFS (очень мутная)

5. У чому смисл глобальної та поточної настройки адресних констант. #const #adr

глобальная – перенастройка всех адресных констант, локальная – вычисление адреса той переменной, которая находится реально в ОП

6. Закон Гроша. #grosh

Производительность компьютера увеличивается как квадрат стоимости. Если компьютер А стоит в два раза дороже, чем компьютер В, то вы должны ожидать, что компьютер А в четыре раза быстрее, чем компьютер В.[1]

7. Як визначити об'єм пам'яті, необхідний для завантаження програми простої структури? #memory #struct

размер самой программы + корневой каталог.

8. Алгоритм Корбато лучше алгоритма FВn, #corbato #fbn #korbato

т.к. потенциально обладает большей производительностью т.к. распределяет задачи по очередям не по их приоритетам, а по признаку - абсолютная длина кода программы (чем меньше код тем выше приоритет)(В систему добавлен анализатор, который сразу размещает заявки в свою очередь, соответственно среднее время ожидания уменьшается.)

9. Де знаходяться нові PSW? #psw #new

Значение нового PSW находится в векторе прерывания с номером n, где n указывает номер прерывания обработчик которого нужно взять

10. Ідея застосування методу оціночних функцій. Переваги й недоліки.

11. Недолік колективного доступу та як він усувається у сучасних системах? #collect #colect #access

Предусматривающий доступ к ресурсам системы_(система работает в многопрограммном режиме) многих пользователей, одновременная работа нескольких userov на машине (Логика прерываний). Служба пропускания не належить одному користувачу, можуть бути проблеми з QoS.

12. Чому в розподілених системах переважніше використовувати крупнозернисте планування? #zern #plan

Крупнозернистое планирование – когда время выполнения узла больше времени пересылки

13. Пояснити термін «дружня операційна система»

14. Функції рівня операційної системи у багаторівневій системі програмування.

15. Описати рівні статичного планування для одно процесорної ОС. #plan #level

Раннее планирование – запускаем задачу, как только есть все условия. Позднее – придерживаем задачу без увеличения критического пути

Білет 13

1. Який тип транслятора використовується при трансляції з традиційної машинної мови у команди машини та як він реалізується? #transl #asm #codes

По идее, Ассемблер. Процесс трансляции программы на языке ассемблера в объектный код принято называть ассемблированием. В отличие от компилирования, ассемблирование — более или менее однозначный и обратимый процесс. В языке ассемблера каждой мнемонике соответствует одна машинная инструкция, в то время как в языках программирования высокого уровня за каждым выражением может скрываться большое количество различных инструкций. В принципе, это деление достаточно условно, поэтому иногда трансляцию ассемблерных программ также называют компиляцией.

2. Визначення метаобчислень (суперобчислень) #metacount #supercount #count

Теория метасистемных переходов - это теория эволюции. Условия реализации эволюции можно создать в мире программ. Достаточно написать для них метапрограммы, то есть программы, преобразующие другие программы. Последовательное и (формально) бесконечное применение метапрограмм к программам и к самим себе будет порождать новые программы. При этом могут возникать метасистемные переходы, приводящие к качественно новым возможностям программ. Созданием метапрограмм (новых метасистем) над программами на основе методов анализа и преобразования программ и занимается теория метавычислений. =))

3. Чому в розподілених системах переважніше використовувати крупнозернисте планування? #plan #zern

Крупнозернистое планирование – когда время выполнения узла больше времени пересылки

4. Дії, що виконує програма ініціалізації ядра. Звідки вона викликається? #kernel #init

При инициализации ядра и системы активизируются и исполняются следующие процессы операционной системы: администратор памяти, программа работы с ВУ, программа поддержки файловой системы и процесс, подчиненный таймеру. Вызывается BIOS=>первичный загрузчик=>вторичный загрузчик=>прогр. Иниц.ядра

5. Функції завантажника, що налаштовує. Яку інформацію повідомляє йому компілятор? #loader #compiler

Распределение памяти, настройка, редактирование, загрузка. Инфу для него готовит компилятор. Настраивающий – работает в загр. Модуле которому больше ничего не надо. Постепенно его ф-ции взял на себя редактор связей. Непосредственно в настраивающем загрузчике каждый модуль может транслироваться отдельно. Чтобы передать сообщение редактору связей надо ему непосредственно указать, что надо транслировать. В каждом модуле в начале трансляции выделяются вектора перехода, внешние и внутренние.(Экспорт и Импорт процедур и ф-ций). Кроме того для выполнения настройки каждая команда отмечается битом переместимости. ОС выделяет и пользуется глобально выделенной памятью, а загрузчик с локальной.

6. Сформулювати теорему про обов'язкові призначення. #prizn #obov

С помощью анализа матрицы выделяют назначение, которое есть обязательным еще до нахождения базового решения. Теорема 1: Если в двудольном графе существует вершина, которой инцидентно только одно ребро, то вершины, инцидентные этому ребру, должны быть взяты в решение

7. Як визначити об'єм пам'яті, необхідний для завантаження програми оверлейної структури? #struct #loader #overlay

? Гляньте ще теги overlay. Корневой сегмент + самый большой модуль.

8. Основна особливість пріоритетних дисциплін обслуговування. Види. #obslug #prior

Приоритетное обслуживание – заявка на вход системы с заданным приоритетом. Относительный приоритет – не может прервать задачу на ресурсе, даже если она имеет низкий приоритет. Абсолютный приоритет – прерывает задачу на ресурсе если та имеет более низкий приоритет. Динамический приоритет – ЕСЛИ возникает опасность бесконечного откладывания. ЕСЛИ по мнению sys, заявка слишком долго занимает ресурс, ТО ее приоритет понижается. ЕСЛИ заявка очень долго ожидает ресурс ТО ее приоритет повышается.(QoS). Если заявка с более высоким приоритетом не прерывает обслуживание заявки с низким приоритетом – относительная ДО без вытеснения, соответственно, если наоборот - абсолютная ДО с вытеснением. Приоритеты бывают: * статические (сразу задаются заявке и не изменяются в процессе) * динамические (приоритеты меняются в зависимости от тожд. или тобслуж.).

9. Дати визначення паралельної системи. #parallel #paralel #sys

Параллельные вычислительные системы — это физические компьютерные, а также программные системы, реализующие тем или иным способом параллельную обработку данных на многих вычислительных узлах.[1]
Вики

10. Ідея розв'язання задачі планування методом «пізнього» планування. #plan #pizn

придерживаем задачу без увеличения критического пути

11. Задача «наповнення рюкзака». Для розв'язання якої задачі планування можна її застосовувати?

12. Яка інформація міститься у MBR? #mbr

код загрузчика, таблицу разделов (partition table) и специальную сигнатуру

13. Дати визначення процесу. Коли це поняття стало застосовуватися й чому? #process #epic #lol

Процесс – это траектория процессора в адресном пространстве машины

14. Застосування GRID систем. #grid #usage

Эта технология применяется для решения научных, математических задач, требующих значительных вычислительных ресурсов. Грид-вычисления используются также в коммерческой инфраструктуре для решения таких трудоёмких задач, как экономическое прогнозирование, сейсмоанализ, разработка и изучение свойств новых лекарств.+адронный коллайдер

15 Відмінність завдання від задачі. Де зберігається завдання? #task

Задание – внешняя единица работы системы (описывается на специальном языке). Задание загружается в систему только тогда, когда система имеет свободные ресурсы и преобразуется в задачу. Задача – внутренняя единица работы системы, для которой система выделила ресурсы кроме процессорного времени. Задача фиксируется в системе если ей выделены системные ресурсы (блок управления процессом).

Білет 14

1. Що таке розв'язання задачі планування з передуманням? #pered #task #plan

Задача после стрелки (на графе, из конспекта)) решается только после того, как решена задача до стрелки. Это при статическом планировании

2. Сформулювати теорему про обов'язкові призначення. #prizn #obov

С помощью анализа матрицы выделяют назначение, которое есть обязательным еще до нахождения базового решения.

Теорема 1: Если в двудольном графе существует вершина, которой инцидентно только одно ребро, то вершины, инцидентные этому ребру, должны быть взяты в решение

3. Яка програма виконує запис інформації в область збереження та яка інформація туди записується? #save #obl #store

Область сохранения - область в которую записывается информация о прерванном процессе, и откуда система берет данные при восстановлении процесса. Какая программа выполняет запись в область сохранения(?) – Та часть ядра которая переключает задачи

4. Обробка станів P3 й P4. #state #p3 #p4

В P3 происходит выполнение процедуры сохранения, дешифрации прерывания, а также восстановления прерванной программы. P4 – состояние, в которое машина входит от схем контроля

5. Який тип завантажника працює з модулями з розширеннями COM та EXE? #com #exe #modul #loader

С COM - абсолютный; с EXE - настраивающий.

6. На яку характеристику впливають пріоритетні дисципліни обслуговування заявок? #prior #obslug

Я думаю, логично предположить, на среднее время ожидания

7. Як визначити об'єм пам'яті, необхідний для завантаження програми динамічної паралельної структури? #memory #dynamic #struct

Корневой сегмент + наибольший суммирующий объем модулей, которые могут выполняться параллельно

8. Математична постановка задачі статичного планування. #plan #math

1. Для задачи, заданной в виде ациклического ориентированного графа, найти минимальное количество процессоров, чтобы время решения не превышало Ткритическое.
2. Для того же графа найти такое распределение задач по процессорам, когда минимизируется время решения при заданном кол-ве процессоров.

9. Чому й коли блокується система переривань? #interrupt #block

При дешифрації текущего прерывания

10 Як змінюється організація обчислювального процесу для систем зі спільною пам'яттю та SMP?

11. Сформулювати теорему про віяло. #japan #vijalo #vijalo #theorem #theorem

Теорема:

Если в двудольном графе есть вершины, которым инцидентно одно ребро, то в этом случае в решение берутся эти вершины, остальное - редуцируется

12. Що таке віртуальна операційна система, чи застосовується вона у сучасних ОС? #os #virt

Так называемая ОС, которая позволяет многим пользователям работающим на одной и той же технической базе(одно ус-во) одновременно работать в различных операционных средах. Виртуальные ОС – разрешает нескольким useram работать на одном и том же dev на разных ОС.

13. Проблеми керування паралельними процесами. Чому вони важкі для реалізації? #process #hard #drive

Добавляется планирование в пространстве; появляется проблема синхронизации процессов. Тутики

14. Для чого слугує характеристика «часова складність алгоритму»? #time #sklad

Я думаю, логично предположить, для оценки и сравнения эффективности работы алгоритмов

15. Що таке «строга неоднорідна ОС»? Дати приклад такої неоднорідності. #neodnor #os

структуры, где узлы сильно разнятся между собой по своим аппаратным ресурсам, программному обеспечению и структурам данных, соответственно одна и та же заявка на разных узлах может быть обслужена с разной скоростью

Білет 15

1. Дати характеристику неоднорідної системи. #neodnor #sys

Для железа – система содержит процесоры разной специализации и/или производительности. Если имеется ввиду ОС – то хз.

2. Процес. Коли та як визначаються пріоритети системних й проблемних процесів?

3. Як передаються дані між модулями в оверлейних системах? #data #module #overlay #sys

(Не уверен, со справочника по Borland Pascal): передача через стек основной программы.

4. Які операції виконує завантажник при завантаженні EXE файлу? #loader #ex

1. В области памяти после резидентной части выполняющей загрузку программы строится Префикс Программного сегмента.
2. Стандартная часть заголовка считывается в память.
3. Определяется длина тела загрузочного модуля (разность длины файла 04-07 и длины заголовка 08-09 плюс число байт в последнем блоке 02-03). В зависимости от признака, указывающего загружать задачу в конец памяти или в начало, определяется сегментный адрес для загрузки. Этот сегмент называется начальным сегментом.
4. Загрузочный модуль считывается в начальный сегмент.
5. Таблица настройки порциями считывается в рабочую память.
6. Для каждого элемента таблицы настройки к полю сегмента прибавляется сегментный адрес начального сегмента. В результате элемент таблицы указывает на нужное слово в памяти; к этому слову прибавляется сегментный адрес начального сегмента.
7. Когда таблица настройки адресов обработана, регистрам SS и SP придаются значения, указанные в заголовке, к SS прибавляется сегментный адрес начального сегмента. В ES и DS засылается сегментный адрес начала Префикса Программного сегмента. Управление передается загруженной задаче по адресу, указанному в заголовке (байты 14-17).

5. Дати визначення резидентного модуля. Коли визначається склад резидентних програм? #resident #module

Резидентная программа (или TSR-программа, от англ. Terminate and Stay Resident — «завершиться и остаться

резидентной») — в операционной системе MS-DOS программа, вернувшая управление оболочке операционной системы (command.com), либо надстройке над операционной системой (Norton Commander и т. п.), но оставшаяся в оперативной памяти персонального компьютера. Резидентная программа активизируется каждый раз при возникновении прерывания, вектор которого эта программа изменила на адрес одной из своих процедур.

6. Дати визначення табличного методу керування. Чому він застосовується?

7. Яка системна програма підготовлює інформацію для роботи завантажника, та яку? Компілятор/линковщик.

8. Як визначити зону знаходження оптимального розкладу? Зі спільною пам'яттю та зі спільною шиною.

9. Структура супервізора.

10. Що таке транзитні програми? Спосіб їх завантаження. #transit #program

Программы, связанные с выполнением функций ОС, но не находящиеся постоянно в ОП называются транзитными. Эти программы вызываются в ОП по мере необходимости.

11. Що таке запис у бібліотеку без каталогізації? Дати приклад. #store #lib #catalog

Запись временных данных, которые удаляются после выполнения программы.

12. Як змінюється інформація системи динамічного планування для однорідних та неоднорідних ОС?

13. Математична постановка задачі планування для одно процесорних систем.

14. Що таке розв'язання задачі планування з передумовинами?

15. Принципи підвищення ефективності ОС.

Білет 16

1. Які міркування використовуються при визначенні величини кванта? #kvant #size #doesmatters

Чем меньше квант – тем больше переключений задач (возможны БОльшие дополнительные расходы). При большом кванте возможна ситуация, когда время работы задачи меньше кванта, тогда оставшееся время процессор простаивает. Также при большом кванте RR, FВп, Корбато очередь стремится к FIFO-виду.

2. Умова переходу процесу з підготованого стану у готовий та з активного в підготований?

3. Чим відрізняється традиційна машинна мова програмування від команд машини? #command #machine #lang

Команды машины - это самые элементарные выполняемые команды (самый низкий уровень); традиционный машинный язык - это набор процедур из элементарных команд, выполняемых на самом низком уровне.

4. Сенс розв'язання задачі реконфігурації при балансовому плануванні.

5. Що таке масштабування розподілених систем?

6. Чому застосовуються багаторівневі системи програмування?

7. Яку інформацію та як компілятор передає завантажнику, що налаштовує.

8. Функції редактора зв'язків. Як йому передається інформація?

9. Як визначити вершини графа, що знаходяться на критичному шляху?

10. Які методи використовуються для розв'язання NP-повних задач? #np #method

* Метод ветвей и границ состоит в отбрасывании заведомо неоптимальных решений целыми классами в соответствии с некоторой оценкой * Метод локальных улучшений состоит в поиске более оптимального решения в окрестности некоторого текущего решения * Приближенные и эвристические методы состоят в применении эвристик для выбора элементов решения * Псевдополиномиальные алгоритмы представляют собой подкласс динамического программирования * Метод случайного поиска состоит в представлении выбора последовательностью случайных выборов

11. У чому відмінність розв'язання задачі планування від розподілення? Яка з них розв'язується при динамічному плануванні?

12. Функція мети задачі динамічного планування.

13. Сформулювати теорему про потужність паросполучення.

14. За якою схемою завантажуються транзитні програми супервізора?

15. Що таке межа Бременмана?

Білет 17

1. Дати визначення непрямого доступу. Чи застосовується він зараз й коли? #per #access

Непрямой доступ – (косвенный доступ) – доступ к машине через управляющую программу (прообраз ОС, пакетный режим)

2. Що таке модульний принцип програмування? #program #module

Модульный принцип программирования – идея: программа разбита на куски. Каждый кусок выполняется с определенными соглашениями.

Свойства:

- каждый модуль обладает стандартностью внутр. Структуры
- взаимная независимость
- функц. Завершенность
- параметрическая универсальность

Виды модулей – вызывающий и вызываемый.

3. Сформулювати принцип оптимальності Белмана. #princ #belman #optim

Принцип оптимальности Беллмана - замена решения исходной многомерной задачи последовательностью задач меньшей размерности.

4. Характеристика розподіленої операційної системи. Чим вона відрізняється від мережної? #net #os #sys #rozpod

Распределенная ОС - совокупность выч. узлов, связанных между собой каналами связи, с точки зрения usera представляют собой единое целое(одна вирт машина). Функции: расширяемость, масштабируемость, прозрачность

Сетевая ОС предоставляет пользователю некую виртуальную вычислительную систему, работать с которой гораздо проще, чем с реальной сетевой аппаратурой. В тоже время эта виртуальная система не полностью скрывает распределенную природу своего реального прототипа, то есть является виртуальной сетью. При работе юзер точно знает на какой машине выполняется задача.

5. Види завантажників та їх особливості. Коли який застосовується? #loader

Функции:

- Выделение памяти
- настройка
- связывание
- погружение

Виды загрузчиков: в зависимости от того, кто выполняет функции(прогр или система) разделяют: - абсолютный загрузчик(выполняет только посл функцию)

- настраивающий (выполняет все функции (работает при создании .exe файла), компилятор ему должен сообщить доп. информацию: внешние и внутренние вектора переходов и биты переместимости (?) для настройки адресных констант)

- связывающий (непосредственно связывающий) – все функции выполняются по мере необходимости (ООП)

6. Які програми знаходяться у ядрі операційної системи? (Види програм) #kernel #os #program

ядро супервізора, включающие только те программы, которые отвечают за реакцию системы.

Гугл:

- обработка прерываний;
- создание и уничтожение процессов;
- переключение процессов из состояния в состояние;

- диспетчеризацию заданий, процессов и ресурсов;
- приостановка и активизация процессов;
- синхронизация процессов;
- организация взаимодействия между процессами;
- манипулирование РСВ;
- поддержка операций ввода/вывода;
- поддержка распределения и перераспределения памяти;
- поддержка работы файловой системы;
- поддержка механизма вызова — возврата при обращении к процедурам;
- поддержка определенных функций по ведению учета работы машины;

7. Дати визначення програми простої структури. Коли використання такої структури неефективно?

#program #simple #struct

Программа которая при её исполнении не обращается к другим программам. Простейшая структура (имеют в своем теле все что нужно для выполнения)

8. Вхідна та вихідна інформація редактору зв'язків. : #editor #link #redak #input #output

vx- объектный модуль , вых – загрузочный модуль.

9. Як визначити вершини графу, що знаходяться на критичному шляху? #graph #versh #detect

(?) это вершины, которые попадают на максимальный путь по графу (худшее время решения задачи)

10. Основна ідея «угорського алгоритму». Заміна якого блоку покращує його характеристики? : #venger #algo #idea

-если из всех элементов некоей строки или столбца вычесть одно и то же число , общая стоимость уменьшится на , а оптимальное решение не изменится;

-если есть решение нулевой стоимости, оно оптимально.

Алгоритм ищет значения, которые надо вычесть из всех элементов каждой строки и каждого столбца (разные для разных строк и столбцов), такие что все элементы матрицы останутся неотрицательными, но появится нулевое решение.

11. Навіщо потрібно водити характеристику «надійний стан системи»? #os #state #nadiyn #nadijn

Надежное состояние системы - ОС может обеспечить всем процессам их выполнение в течение конечного времени.

12. Ідея алгоритмів, заснованих на стратегії критичного шляху #crit #idea #algo #strateg

Основная идея – кластеризация. Критический путь грузится на 1 проц и все пересылки зануляются.

13. Методи підвищення ефективності організації обчислювального процесу в ОС. #improve #better #effect #os #count

1. Декомпозиция задачи.
2. Методы перебора состояний
3. Приведение целевой функции к сепарабельному виду.
4. Распараллеливание вычислительного процесса.
5. Разработка квазиоптимальных графиков.

14. Що таке шлях, що збільшує, шлях, що чергується? #way #path #goto #whoison dutytoday

(?)Теорема Герта: тах парасочетания получают, если от свободной вершины нельзя найти увеличивающийся путь.Идея: для двудольного графа, случайно, выделяются ребра, которые не имеют инцендентных вершин.

Пространственный планировщик – задача поиск максимального парасочетания в двудольном графе. Вот там и используется увеличивающийся чередующийся путь.

15. Які задачі розв'язуються при статичному плануванні у паралельній ОС? #os #static #plan

Стат планирование: NP-полные задачи, только для специализированных систем. 2 типа: минимальное время решения при заданном кол-ве проц; минимальное количество процессоров, чтобы время не превышало Tmax.

Білет 18

1. Сформулювати принцип оптимальності Белмана. #belman #optim #princ

Принцип оптимальности Белмана - замена решения исходной многомерной задачи последовательностью задач

меншій розмірності.

2. Чи використовується у сучасних системах колективний доступ та чим він відрізняється від того, що використовувався раніше? #collect #access

Колективний доступ- Передбачаючий доступ к ресурсам системи (система працює в багатопрограмному режимі) багатьох користувачів. Відмінності (?)

3. Умови створення процесу. Який ресурс при цьому є критичним? #process

Коректність описання процесу на мові описання процесів і виділення ресурсів Критичний ресурс – адресне простір

4. Умови переходу зі стану P1 у P2 та назад. #p1 #p2 #p2reh

P1- системні процеси, P2- користувацькі. У системних пріоритет вище. Переходи по квантах.

5. Основні недоліки застосування мультипрограмування. #multi #program #bad

Недостатки пов'язані з розподілом пам'яті: надмірне виділення, фрагментація

6. Чому у розподілених системах переважніше використовувати крупнозернисте планування? #zern #plan

Крупнозернисте планування – Виконання << Тривалості (?)

7. Дати визначення програми оверлейної структури. Як передаються дані? #overlay #struct #data

Програма оверлейної структури -Задача, розділяється на модулі. Модулі знаходячись на одному оверлейному рівні не можуть одночасно знаходитись в ОП. Оверлей (задані перекриття двох модулів на одному оверлейному рівні не можуть бути виконані, Інфа передається via кореневої модуль)

8. Сформулювати теорему про конфліктні призначення. #theorem #theorem #conflict

ЕСЛИ в матриці зв'язності можна виділити основну підматрицю (з нулів) $x \times y$, $x+y=N$ і розмістити її в верхньому правому куті, тоді всі одиниці входять в матрицю симетричну їй відносно головної діагоналі являються конфліктними і повинні бути видалені з розгляду(анульовані) -> граф редукується => граф розпадається на дві частини і пошук паросочетаній може вестись окремо. (ЕСЛИ декілька основних підматриць ТО граф розпадається на декілька частин)

9. Дати визначення розподіленої операційної системи. Її відмінність від мережної. #net #distrib #os

Розподілена ОС - сукупність вузлів, зв'язаних між собою каналами зв'язу, з точки зору користувача представляють собою єдине ціле(одна віртуальна машина). Функції: масштабованість, прозорість

Мережова ОС надає користувачеві певну віртуальну обчислювальну систему, працювати з якою значно простіше, ніж з реальною мережею апаратури. В той же час ця віртуальна система не повністю приховує розподілену природу свого реального прототипу, тобто є віртуальною мережею. При роботі користувач точно знає на якій машині виконується задача.

10. Ідея створення адаптивних алгоритмів планування. (?) Система аналізує штраф за час очікування заявки і штраф за час обслуговування.

11. Особливості різних форм взаємодії «людина – машина». Які характеристики при цьому змінюються?

12. У чому складність для операційної системи в організації багатопрограмного режиму роботи? Які задачі при цьому розв'язуються та які механізми використовуються? #os #multi #prog #meh

Організація захисту від взаємного впливу однієї на іншу на рівні оперативної і на рівні зовнішньої пам'яті; розподіл апаратних і програмних ресурсів; планування (по часу, а в разі ПВС і в просторі). Створюється система переривань, система управління процесами.

13. В якому вигляді компілятор передає інформацію завантажнику, що налаштовує? #compiler #loader #data

В вигляді структури: зовнішні і внутрішні вектори переходів; біти перемістимості для налаштування адресних констант.

14. Перерахувати рівні планування у паралельній системі. Які задачі потрібно розв'язувати у статичній, а які – у динамічній? #plan #level #sys #parallel #parallel

Динамічне, статичне, балансове, в реальному часі. В динаміці – робота по часу рішення задачі, на той же обладнання, швидка обробка з ушкодженням якості; в статичній – планування рішення задачі ДО самого рішення на іншому обладнанні.

15. Що таке «строго неоднорідна ОС»? Які параметри ОС визначають неоднорідність? #neodnor #sys #param
Є вище

Білет 19

1. Перерахувати способи реалізації багатопрограмного режиму роботи. #multi #program #work

Классическое мультипрограммирование (пакетная обработка)

Разделение времени

Режим реального времени

2. Сформулювати теорему про конфліктні призначення. #conflict #theorem #theorem

ЕСЛИ в матрице связности можно выделить основную подматрицу (из нулей) $x \cdot y$, $x + y = N$ и расположить ее в верхнем правом углу, тогда все единицы входящие в матрицу симметричную ей относительно главной диагонали являются конфликтными и должны быть удалены из рассмотрения (занулены) \rightarrow граф редуцируется \Rightarrow граф распадается на две части и поиск парасочетаний может вестись отдельно. (ЕСЛИ несколько основных подматриц ТО граф распадается на несколько частей)

3. Вимоги до статичних планувальників. #static #plan

Годится только для спец. систем, NP-полных задач, в статике – планирование решения задач ДО самого решения на другом оборудовании.

2 типа: минимальное время решения при заданном кол-ве проц; минимальное количество процессоров, чтобы время не превышало T_{max} .

4. Чим визначається пріоритет проблемних й системних програм та коли ці пріоритети визначаються?

#prior #problem #sysem

Приоритет системных программ выше приоритета проблемных. (?)

5. Перерахувати ресурси ОС. : #os #res

железо(всё); программы, информация.

6. Дати визначення резидентного тому. Яка системна програма формує зміст резидентного тома? #tom

#resident

Том – место нахождения резиденции – сгенерированной ОС. Формируется программой начальной загрузки(БУТлоадер) которая нах. В MBR.(?)

7. Дати визначення програми динамічної послідовної структури. #struct #dynamic Програми настроенные с помощью модульного принципа, представляется в перемещаемом виде, могут подгружаться по мере необходимости с организацией связей по управлению и данными. 3. Динамически посл.(Память выделяется по мере надобности – по цепочке. Уровни не выделяются. Корневой модуль всегда в памяти)

8. Різниця між обчислювальною та часовою складністю. Яка оцінка коли застосовується? #time #count

#skaldn

Временная сложность – зависимость времени решения задачи от размерности задачи.(худший случай)

Вычислительная – зависимость объема работы от размера входных данных. Когда применяется (?)

9. Функція мети задачі статичного планування. Чому потрібно розділяти планування та розподілення?

#meta #static #plan

Годится только для спец. систем, NP-полных задач, в статике – планирование решения задач ДО самого решения на другом оборудовании.

2 типа: минимальное время решения при заданном кол-ве проц; минимальное количество процессоров, чтобы время не превышало T_{max} .

Разделять нужно потому что выполняются на разных машинах (оборудовании)

10. Динамічне планування для систем реального часу. . #dynamic #plan

Время реакции системы соответствует ранее заданному.

Т работы Приоритеты задачам распр динамически – предпочтение задачам с меньшим Т работы.

11. Дати визначення керуючих програм операційної системи. #program #os #ker #boss

Программы постоянно находящиеся в памяти (резидентные) организующие корректное выполнение процессов и функционирование всех устройств системы при решении задач. Составляют ядро ОС. Управление : Заданиями – слежение за прохождением заданий от входа до выхода на всех этапах его выполнения. Задачами – (Процессами)

– слежение за всеми задачами активизированными в системе и процессами их выполнения на ресурсах. Памятью – решение задач эффективного и/ mem (internal) в соответствии с ее организацией. (Защита – согласование ин-фы в кешах) Данными – эффективное размещение и/ данных на внешних носителях (проблема эффективности и/ процессора) Внешними ус-вами ...

12. Що таке пряме введення-виведення? Чому непряме краще? #іо

Прямой : Когда не используются управляющие программы. Пользовательский процесс сам осуществляет і/о. Непрямой лучше потому что обеспечивается ОС(а значит надежней, правильной, круче, выше, дальше, смелее, добрее, няшней)

13. Як компілятор передає інформацію безпосередньо завантажнику, що зв'язує? #compiler #loader

Настраивающий – работает в загр. Модуле которому больше ничего не надо. Постепенно его ф-ции взял на себя редактор связей. Непосредственно в настраивающем загрузчике каждый модуль может транслироваться отдельно. Чтобы передать сообщение редактору связей надо ему непосредственно указать, что надо транслировать. В каждом модуле в начале трансляции выделяются вектора перехода, внешние и внутренние. (Экспорт и Импорт процедур и ф-ций). Кроме того для выполнения настройки каждая команда отмечается битом переместимости. ОС выделяет и пользуется глобально выделенной памятью, а загрузчик с локальной.

14. Обчислювальні системи реального часу. Визначення. Особливості #system #realtime

Системы реального времени - режим работы автоматизированной системы обработки информации и управления, при котором учитываются ограничения на временные характеристики функционирования. Если выход за рамки временных ограничений – отказ системы (жесткая СРВ) или ухудшение качества (мягкая СРВ). Системы реального времени обычно используют специализированное оборудование и программное обеспечение. При создании систем реального времени приходится решать проблемы привязки внутрисистемных событий к моментам времени, своевременного захвата и освобождения системных ресурсов, синхронизации вычислительных процессов, буферизации потоков данных и т. п.

15. Назвати нижні рівні керування зовнішніми пристроями. #lover #level #drive #manage #pristr #device

Нижние уровни управления внешними устройствами (?)

Білет 20

Білет 21

1. Сформулювати принцип оптимальності Белмана. #bellman #belman #princ #optim

БЕЛЛМАНА ПРИНЦИП ОПТИМАЛЬНОСТИ [Bellman's optimality principle] — важнейшее положение динамического программирования, которое гласит: оптимальное поведение в задачах динамического программирования обладает тем свойством, что каковы бы ни были первоначальное состояние и решение (т. е. “управление”), последующие решения должны составлять оптимальное поведение относительно состояния, получающегося в результате первого решения. Этот принцип можно выразить и рассуждая от противного: если не использовать наилучшим образом то, чем мы располагаем сейчас, то и в дальнейшем не удастся наилучшим образом распорядиться тем, что мы могли бы иметь. Следовательно, если имеется оптимальная траектория, то и любой ее участок представляет собой оптимальную траекторию. Этот принцип позволяет сформулировать эффективный метод решения широкого класса многошаговых задач. (Подробнее см. Динамическое программирование.)

2. Коли застосовують евристичні алгоритми? #evrist #algo

В інформатиці, евристичний алгоритм, або просто евристика, це алгоритм спроможний видати прийнятне рішення проблеми серед багатьох рішень, але неспроможний гарантувати, що це рішення буде найкращим. Отже такі алгоритми є приблизними і неточними. Зазвичай такі алгоритми знаходять рішення близьке до найкращого і роблять це швидко. Часто, можна знайти таку задачу, що евристичний алгоритм буде працювати або дуже довго, або видавати невірні результати, однак, такі парадоксальні приклади можуть ніколи не зустрітись на практиці через свою специфічну структуру. Таким чином, використання евристики дуже поширене в реальному світі. Для багатьох практичних проблем евристичні алгоритми, можливо, єдиний шлях для отримання гарного рішення в прийнятний проміжок часу. Існує клас евристичних стратегій названих метаалгоритмами, котрі часто використовують, наприклад, випадковий пошук. Такі алгоритми можуть бути застосовані до широкого кола завдань, при цьому хороші характеристики не гарантуються.

3. Яка програма виконує запис інформації в область збереження та що там зберігається. #write #info #save

Ядро ОС. Записываются значения регистров при переключении процессов из состояния в состояние либо их синхронизации.

4. Що таке резидентні вершини та як їх знайти? Навіщо вони потрібні? #resident #versh

Вершини, що увійшли в критичний шлях, а їх часові та просторові координати не повинні змінюватись при вирішенні задач. Знайти критичний шлях.

5. Функції завантажника, що налаштовує. #set #loader

Настраиваемый - До этапа выполнения программы в модулях описываются векторы переходов (внутри – и внешнемодульные), и константы, которые нужно переопределить. Загрузчик при просмотре этих записей определяет абсолютные адреса гастраниваемых величин. Настройка адресных констант, Выделение памяти, Связывание, Собственно загрузка. Инфу для него готовит компилятор.

- выделение места для программ в памяти (распределение);
- фактическое размещение команд и данных в памяти (загрузка); - разрешение символических ссылок между объектами (связывание);
- настройка всех величин в модуле, зависящих от физических адресов в - соответствии с выделенной памятью (перемещение);
- передача управления на входную точку программы (инициализация).

6. Дати визначення ініціалізації системи. Функції. #init #system

В виду того, что ядро ОС представляет собой совокупность взаимосвязанных модулей, имеющих связь друг с другом, процесс структурной организации модулей называется инициализацией. Инициализация системы - процесс создания ядра системы который включает не только перезапись программ, но и системных структур, обеспечивающих знания системы о ее параметрах. Программа - загрузчик. Small intro: ОС предназначена для удовлетворения нужд пользователей и должна содержать все необходимые пользователю модули или программы и поддерживать аппаратные средства. Инициализация – процесс структурной организации взаимосвязанных модулей, из которых состоит ядро ОС. Различают инициализацию ядра ,которая происходит по умолчанию и системы ,которая происходит по файлам инициализации.

7. У чому відмінність між максимальним паросполученням та досконалим. #max #para #doscon #ideal

Найти максимальное паросочетание - значит найти максимальное число ребер графа в которых не совпадают координаты вершин, либо найти максимальное число единиц матрицы у которых не совпадают координаты.

8. Чому алгоритм «Корбато» краще FBn. #corbato #algo #corbato #fbn

Алгоритм Корбата лучше алгоритма FBn, т.к. потенциально обладает большей производительностью т.к. распределяет задачи по очередям не по их приоритетам, а по признаку - абсолютная длина кода программы (чем меньше код тем выше приоритет). (В систему добавлен анализатор, который сразу размещает заявки в свою очередь, соответственно среднее время ожидания уменьшается.)

9. Що таке неоднорідна система? Як визначити степінь неоднорідності. #sys #neodnor

Вычислительную систему можно рассматривать как совокупность неоднородных ресурсов с различной степенью детализации, а объекты планирования вычислений или заявки на ресурсы могут быть представлены как программы, процедуры, параллельные участки программ и т.д.

Понятие "неоднородность" системы можно рассматривать для трех случаев:

1 В полунеоднородной системе, когда или задания, или ресурсы являются однородными или неоднородными. Число различных расписаний (до оптимизации) равно размещению из R элементов по N без повторения и без перестановки, то есть число возможных вариантов равно [21, 59, 134].

2 В неоднородной системе и задания, и ресурсы являются неоднородными, и любое назначение задание→ресурс всегда является возможным (лишь имеют разные значения Те и Тс). Число различных расписаний (до оптимизации) равно размещению из R элементов по N без повторения, но с перестановками, тогда число возможных вариантов равно [72].

В строго неоднородной системе имеет место понятие совместимости ресурса→задания, при этом некоторые назначения ресурс→задание могут быть невозможными. Поэтому планирование для таких систем состоит из двух этапов: на первом этапе ищутся расписания возможных назначений максимальной мощности, а на втором этапе требуется найти оптимальное расписание из найденных на предыдущем этапе. Число возможных вариантов на первом этапе равно числу различных и возможных расписаний (до оптимизации), или равно числу максимальных паросочетаний двух множеств N и R $V_1 \leq [13, 40]$. Число возможных вариантов на втором этапе $V_2 \leq V_1$.

10. Функція мети задачі динамічного планування. #dynamic #plan #meta

Под динамическим планированием понимается решение задачи распределения работ (составление расписания) и

процессов в пространственных или пространственно-временных координатах во время выполнения вычислений и на том же оборудовании. Если планирование составляется на том же оборудовании на котором должно выполняться и во время решения задачи, то это планирование называется динамическим

11. Сформулювати теорему про обов'язкові призначення. #theorem #theorem #obov #prizn

С помощью анализа матрицы выделяют назначение, которое есть обязательным еще до нахождения базового решения. Теорема 1 :

Если в двудольном графе существует вершина, которой инцидентно только одно ребро, то вершины, инцидентные этому ребру, должны быть взяты в решение

12. Що таке «свопінг»? #swap

Свопинг - способ реализации многопрограммного режима работы на однопроцессорной машине

13. Різниця між плануванням та розподіленням. #plan

Задача \square распределение (загрузка) (Task Allocation). Вычислительные процессы имеют слабые требования по предшествованию и отображаются несвязными графами. Это направление часто связывается с задачами балансирования и минимизации пересылок. При решении этих задач основой является определение \square какой процесс (задание) будет выполняться на каком процессоре (ресурсе), а не определение порядка их выполнения [155]. При этом, в основном, решаются задачи пространственного распределения процессов (заданий).

Задача \square планирование (Task Scheduling). Вычислительные процессы имеют сильные требования по предшествованию и отображаются связными ациклическими графами (DAG). При этом, к решению задач первого направления добавляется также определение порядка выполнения заданий. На этом уровне часто решаются задачи минимизации суммарного времени выполнения полного DAG на выделенных или имеющихся ресурсах. Задачи по обоим направлениям, в общем случае, являются NP-полными или NP-сложными даже для двухпроцессорной системы [11] и имеют экспоненциальную временную сложность

14. По який характеристиці можна судити про придатність розробленого алгоритму для роботи у динамічному режимі? #dynamic #algo

В большинстве случаев разработчики систем планирования реального времени используют статические алгоритмы и заранее определяют максимальный список заданий, допустив наихудший случай для получения статической управляющей таблицы (плана). Этот план фиксируется и используется для безусловного исполнения в динамическом режиме со следующими допущениями:

3 все временные ограничения остаются неизменными на время выполнения плана;

4 все задачи вкладываются в свое критическое время.

В других случаях при помощи приемов статического планирования создается статический список приоритетов для использования во время диспетчеризации самих работ в динамическом режиме.

15. Етапи завантаження операційної системи. #os #load #boot #zavant #etap

Першим завантажується BIOS. Він тестує обладнання, завантажує таблиці переривань. Завантажується завантажувач MBR, який визначає кількість розділів, чи є на них ОС та яка. Потім загрузка передається ОС. Визначаються апаратні засоби, обирається конфігурація. Після цього відбувається загрузка ядра, його ініціалізація. Потім ініціалізація системи.

Білет 22

1. Дати визначення режиму розподілення часу. #rozpod #time

Реж. раз. Времени - режим совмещающий мультипрограммирование и параллельную обработку, плюс возможность привелигированным пользователям иметь прямой доступ к ресурсам системы

2. Особливості розподілених операційних систем. #rospod #distributive #system #os

Розподілена ОС – сукупність обчислювальних вузлів, з'єднаних між собою каналами зв'язку, з точки зору користувача представляє собою єдине ціле. Відсутність спільної пам'яті призводить до неможливості визначення загального стану за допомогою множини спільних змінних, а неможливість спільного звернення до пам'яті та різниця в затримках передач повідомлень призводить до того, що при визначенні стану будь-якого елементу системи з різних точок можна отримати різні результати. Виконання роботи розподіляється у вузлах, виходячи з міркувань пропускної здатності усієї системи. Розподілені системи мають високий рівень організації паралельних обчислень.

3. Зв'язок модулів по керуванню. Які операції виконуються та якими програмами. #link #module #manage

Связь по данным может быть через общие ресурсы либо через адрес списка параметров. Спомощью макрокоманд

возможно указание:

1 возврат из (i+1)-го модуля только в i-й;

2 возврат из (i+n)-го модуля в i-й;

3* вызывающий модуль после передачи управления стирается Для связи по управлению данными определены некоторые ресурсы

4. Принципи підвищення ефективності роботи системи з допомогою організації обчислювального процесу.

#effect #princ #high #count

Максимальное эффективное использование ресурсов методом эффективного планирования ВП. На однопроцессорных комплексах используется планирование во времени. На многопроцессорных - планирование как в пространстве так и во времени.

5. Який тип завантажника працює з модулями з розширенням COM та EXE. #loader #com #exe

С com модулями работает абсолютный загрузчик, с exe - настраивающий

6. Різниця між плануванням й розподіленням. #plan #rozpod

Задача □ распределение (загрузка) (Task Allocation). Вычислительные процессы имеют слабые требования по предшествованию и отображаются несвязными графами. Это направление часто связывается с задачами балансирования и минимизации пересылок. При решении этих задач основой является определение □ какой процесс (задание) будет выполняться на каком процессоре (ресурсе), а не определение порядка их выполнения [155]. При этом, в основном, решаются задачи пространственного распределения процессов (заданий). Задача □ планирование (Task Scheduling). Вычислительные процессы имеют сильные требования по предшествованию и отображаются связными ациклическими графами (DAG). При этом, к решению задач первого направления добавляется также определение порядка выполнения заданий. На этом уровне часто решаются задачи минимизации суммарного времени выполнения полного DAG на выделенных или имеющихся ресурсах. Задачи по обоим направлениям, в общем случае, являются NP-полными или NP-сложными даже для двухпроцессорной системы [11] и имеют экспоненциальную временную сложность

7. Як визначити об'єм пам'яті, необхідний для завантаження програми оверлейної структури. #memory

#overlay #program

Определить объём памяти, необходимый для загрузки оверлейной структуры можно путём вычисления ОП, необходимой для самой длинной цепочки вызовов

8. Основна особливість пріоритетних дисциплін обслуговування. Їх види #prior #task

Каждая заявка имеет приоритет при входе в систему. Заявки с более высоким приоритетом получают возможность быстрее выйти из системы. В системах со статическими приоритетами возможно бесконечное откладывание. Применяют также системы со смешенными алгоритмами. Бывает абсолютный, относительный, динамический.

9. Ідея оптимізації «базового» рішення. _ #base #optimiz

«Базовое» решение оптимизируется по необходимому критерию - время выполнения, количество процессоров

10. Що таке конфліктне призначення? Навіщо потрібно його виділяти. _ #conflict

Конфликтное назначение уменьшает мощность паросочетания. Это назначения стоящие под главной диагональю после выделения основных подматриц

11. Що таке тимчасова бібліотека? Коли вона використовується #temp #lib

Временная библиотека существует только во время выполнения программы. Не помещается в каталог библиотек.

12. Які операції виконує програма першого рівня планування. #operation #program #level #plan

В интерпретаторе настройка адресных констант происходит в процессе выполнения

13. Назвати рівні багаторівневих систем програмування #level #system #program

Визуальные средства

ЯВУ

Уровень ОС

Ассемблер

Микропрограммы

Процессор

14. Призначення операційних систем #os #usage

ОС предназначена для : 1- нахождения задач в ВС 2- обеспечение максимальной эффективности использования ресурсов ВС

15. Задачі планування в системах масового розпаралелювання та чим вони відрізняються від задач планування для розподілених систем. #plan #system #parallel #paralel

В SMP решаются следующие задачи планирования Вычислит Процессы: 1 - найти минимальное к-во процессоров, когда время выполнения взаимосвязанных задач не превышает Ткр 2- найти минимальное время выполнения комплекса ВС на заданном к-ве процессоров. При этом, если учитывается вес пересылок, нужно обеспечить их минимум

Білет 23

Білет 24

Білет 25

1. Дати визначення роботи в реальному часі. Надати приклад. #real #time #mode

Режим реального времени – время реакции системы соответствует ранее заданному

2. Різниця між плануванням і розподілом.

3. Чим відрізняється традиційна машинна мова програмування від команд машини? #machine #command #lang

Команды машины - это самые элементарные выполняемые команды (самый низкий уровень); традиционный машинный язык - это набор процедур из элементарных команд, выполняемых на самом низком уровне.

4. У чому полягає сенс перегляду команд вперед і коли цей прийом неефективний? #select #comand #prev

Предварительная выборка (увеличивается быстродействие)

5. Відмінність завантажувального модуля від абсолютного. Яка програма виконує перетворення? #modul #loader #abs

Загрузочный модуль имеет все для своего выполнения . Написан на машинно-ориентированном языке, но быть выполненным не может. Нужно настроить адресные константы. Абсолютный – все адресные константы уже настроены.

6. Які операції виконує програма другого рівня планування?.

7. Дати визначення програми оверлейной структури. Недоліки. #overlay #struct

Задача, разделяемая на модули. Модули находящиеся на одном оверлейном уровне не могут одновременно находится в ОП. Оверлей (заданные перекрытия два модуля на одном оверлейном уровне не могут быть выполнены Инфа передается via корневой модуль)

8. Вхідна і вихідна інформація редактора зв'язків. #input #output #info #edit #link

Входна - объектный модуль, выходная - загрузочный модуль.

9. Динамічне планування для систем реального часу. #dynamic #plan #sys #realtime

План составляется на том же оборудовании на котором выполняется решение задач во времени. Имеет ограничение на время составления плана. Динамическое планирование – задача планирования решается на том же оборудовании, что и выполняется план, который она составляет. Быстрое (чтобы не грузить оборудование) и неоптимальное решение Def. Джонсона – найти план распределения по ресурсам, при котором время решения не превышало бы критического с min кол-вом процов. Найти min кол-во ресурсов, чтобы задача решалась за min время. Для 2-х процов задача решается точно. Для одного в условиях RealTime.

10. Як визначити обов'язкове призначення? Навіщо це потрібно робити?

11. Що таке «розширення» розподілених систем? #sys #distrib

Распределенная sys – совокупность выч. узлов, связанных между собой каналами связи, с точки зрения usera представляют собой единое целое. Отсутствие общей памяти приводит к невозможности определения общего состояния с помощью множества совместных переменных, а невозможность совместного обращения к памяти и различие в задержках передач сообщений приводит к тому что при определении состояния какого либо элемента системы из двух различных точек можно получить разные результаты. Выполнение работы распределяется в

узлах исходя из соображения пропускной способности всей системы. Распределенные системы имеют высокий уровень организации параллельных вычислений.

12. Що таке «свопінг», чим відрізняється від «спулінга»? #swap

Свопинг - способ реализации многопрограмного режима работы на однопроцессорной машине.

13. Основні проблеми розв'язування задач планування в багатопроцесорних паралельних системах.

#problem #task #multi #proc

Добавляется планирование в пространстве; появляется проблема синхронизации процессов.

14. Описати послідовність дій, які виконуються завантажувачем при завантаженні програм.

В MBR. Активизация программ начальной загрузки ОС (брехня?)

15. Чому СОМ файли мають обмеження за розміром?

Білет 26

1. На що впливає введення поняття «надійний стан системи»? #sys #nadiyn

Необходимо чтобы система была в надежном состоянии (ОС должна обеспечить всем текущим пользователям (процессам) завершение их заданий в течении конечного времени)

2. Пропозиції Лебедева по підвищенню ефективності роботи ОС та як вони реалізовані в сучасних ОС.

3. Відмінність функцій програм системного введення /виведення від режиму спулінга. #system #spul

Режим буферизации для выравнивания скоростей при вводе и считывании информации из буфера. При работе программ системного ввода и/режим спуллинга – согласование скоростей на входе и выходе.

4. Яка характеристика ОС покращується в змішаних дисциплінах обслуговування? #disc #os #obsl

Дисциплина обслуживания заявок – правило по которому решается задача обработки очереди заявок к каждому ресурсу. Бывают линейные (Fifo, Lifo, Random (нет одинаковых ресурсов)) циклические (RR, FBn, смешанная)

5. Частиною якої програми є редактор зв'язку? #edit #link

Редактирование связей. (Выполнение связывания подпрограмм являющихся внешними по отношению к загружаемому модулю). (С помощью редактора связи мы получаем из объектного модуля загрузочный модуль имеющий всё для своего исполнения).

6. Що таке супервізор й де він знаходиться?. #supervis #mode

Режим супервизора — привилегированный режим работы процессора, как правило используемый для выполнения ядра операционной системы.

7. Навіщо потрібно виконувати структурний аналіз вихідного графа? #graph #analys

Структурный анализ заявок и определение возможности распараллеливания (Задача распараллеливания)

8. Функції 3-го рівня багаторівневих систем програмування.

9. Назвати рівні планування для багатопроцесорних ОС. Функції кожного. #level #proc #multi #os

Семиуровневая модель – схема системы планирования с учетом (Масштабируемость, Разделяемость, Параллельность)

1. Предварительное (входное) планирование исходного потока заявок (задача фильтрации)

2. Структурный анализ взаимосвязи входного потока заявок по ресурсам с определением общих ресурсов (Анализ)

3. Структурный анализ заявок и определение возможности распараллеливания (Задача распараллеливания)

4. Адаптация распределения работ соответственно особенностям ВС (Задача адаптирования)

5. Составление плана – расписания выполнения взаимосвязанных процедур. Оптимизация плана по времени решения и кол-ву ресурсов (Задача Оптимизации)

6. Планирование потока задач претендующих на захват времени процессора на каждый процессор – задача распределения.

7. Выделение процессорного времени, активизация задач. Перераспределение работ в ВС, при отказе оборудования (задача распределения – перераспределения).

10. Коли застосовується запис до бібліотеки без каталогізації? #write #lib #catalog

Запись временных данных, которые удаляются после выполнения программы.

11. Як визначити пріоритет процесу системного і проблемного? #prior #system #problem

Добавляется планирование в пространстве; появляется проблема синхронизации процессов. ЕСЛИ в системе нет одинаковых устройств то очереди обрабатываются по линейной схеме => планирование только во времени ИНАЧЕ планирование во времени и в пространстве.

12. Як змінюється організація обчислювального процесу для SMP і PC?

13. Функція мети завдання динамічного планування. #dynamic #plan

План составляется на том же оборудовании на котором выполняется решение задач во времени. Имеет ограничение на время составления плана. Динамическое планирование – задача планирования решается на том же оборудовании, что и выполняется план, который она составляет. Быстрое (чтобы не грузить оборудование) и неоптимальное решение Def. Джонсона – найти план распределения по ресурсам, при котором время решения не превышало бы критического с min кол-вом процов. Найти min кол-во ресурсов, чтобы задача решалась за min время. Для 2-х процов задача решается точно. Для одного в условиях RealTime.

14. Ідея побудови «базового» рішення.

15. Сформулюйте принцип оптимальності Белмана. #woot

???? Эвристические (основанные на опыте людей), Пошаговое конструирование – разделение всего процесса решения на определенное кол-во шагов и нахождение оптимального или квазиоптимального решения. ЭТО – соответствует принципу оптимальности Белмана.

Білет 27

1. Основні принципи організації обчислювального процесу, що підвищують ефективність роботи ОС.

2. Дати визначення режиму поділу часу. #time #podil

режим совмещающий мультипрограммирование и параллельную обработку, плюс возможность привилегированным пользователям иметь прямой доступ к ресурсам системы. (Логика прерываний)

3. Назвати базові рівні в багаторівневих системах програмування. #level #system

Семиуровневая модель – схема системы планирования с учетом (Масштабируемость, Разделяемость, Параллельность)

1. Предварительное (входное) планирование исходного потока заявок (задача фильтрации)
2. Структурный анализ взаимосвязи входного потока заявок по ресурсам с определением общих ресурсов (Анализ)
3. Структурный анализ заявок и определение возможности распараллеливания (Задача распараллеливания)
4. Адаптация распределения работ соответственно особенностям ВС (Задача адаптирования)
5. Составление плана – расписания выполнения взаимосвязанных процедур. Оптимизация плана по времени решения и кол-ву ресурсов (Задача Оптимизации)
6. Планирование потока задач претендующих на захват времени процессора на каждый процессор – задача распределения.
7. Выделение процессорного времени, активизация задач. Перераспределение работ в ВС, при отказе оборудования (задача распределения – перераспределения).

4. Визначити умови переходу зі стану P3 в стан P й P2.

Дивіться зверху за тегами.

5. Перерахувати пріоритетні дисципліни обслуговування. Яка характеристика впливає на вибір дисципліни?

Заявки имеют приоритет. Дякую, кеп :)

6. Види завантажувачів та їх основні відмінності. #loader

Отличаются выполнением основных 4-х функций: распределения памяти, настройки, редактирования и загрузки. Загрузчики делятся: Абсолютный загрузчик, настраивающий загрузчик, непосредственно - связывающий загрузчик.

7. Чому спулінг схильний до тупиків? #spuling #spooling #tupic #sssr

- 1) Условие взаимного исключения;

- 2) Условие ожидания;
- 3) Условие перераспределения;
- 4) Круговая цепь ожидания.

8. Чому всі алгоритми, засновані на теоремі Бержа про максимальне паросполучення, не придатні для використання в динамічних планувальниках?

9. Особливості розподілених операційних систем. У чому їх відмінність від мережних? #distr #sys #net

Распределенная sys – совокупность выч. узлов, связанных между собой каналами связи, с точки зрения usera представляют собой единое целое. Отсутствие общей памяти приводит к невозможности определения общего состояния с помощью множества совместных переменных, а невозможность совместного обращения к памяти и различие в задержках передач сообщений приводит к тому что при определении состояния какого либо элемента системы из двух различных точек можно получить разные результаты. Выполнение работы распределяется в узлах исходя из соображения пропускной способности всей системы. Распределенные системы имеют высокий уровень организации параллельных вычислений.

10. Види модулів. Участь системних програм в перетворенні модулів.: #module #type #sys

- 1) исходный (текст программы);
- 2) объектный;
- 3) загрузочный;
- 4) исполняемый (абсолютный).

Системные программы (компилятор, редактор связей, загрузчик) используются для преобразования от 1) к 4).

11. Що таке обчислювальна складність? #skl #obch #bigo #obig

вычислительная сложность алгоритма — это функция, определяющая зависимость объёма работы, выполняемой некоторым алгоритмом, от размера входных данных. Раздел, изучающий вычислительную сложность, называется теорией сложности вычислений.

12. Основна ідея «угорського алгоритму». #venger #ugor #algo

алгоритм оптимизации, решающий задачу о назначениях за полиномиальное время

13. Чому не можна зациклити стан P3 й P4?

Є зверху

14. Коли застосовують евристичні алгоритми?.

Є зверху

15. Яким чином і коли фіксується пріоритет процесів?

Білет 28

1. Чому впровадження логіки переривань зумовило підвищення продуктивності? #logic #interrupt #process

Якщо не було б переривань. То потрібно було б завжди робити циклічний опрос системи, а на це витрачається багато часу, тому швидше використовувати логіку переривань, яка єдиноразово при виклику видасть те, що нам потрібно.

2. Що таке максимальне паросполучення? #max #para

Максимальне паросполучення - це максимально можлива кількість ребер, які мають спільних кінців.

3. Недолік непрямого методу доступу. Де і як зараз він використовується? #access #perg

Режим непрямого доступу має істотний недолік. Він не дозволяє повністю виключити випадки простою процесора або непродуктивного його використання. Всякий раз, коли чергова програма, викликана в процесор, попередньо не забезпечена даними, процесор змушений простоювати. При цьому різко знижується ефективність використання ЕОМ.

Використовуються, наприклад в SQL-запросах

4. Дати визначення завдання, програми, даних. #data #task #program

Завдання – це одиниця, для якої виділяються ресурси.

Програма – це послідовність, у тому числі кількох паралельних, які виконуються ЕОМ для досягнення поставленої мети або завдання.

(Інше визначення: Програма – це набір інструкцій, які деталізують обчислення або завдання) Дані – це відомості,

факти, показники, виражені в числовій так і в будь-якій іншій формі.

5. Дати визначення ресурсу в сучасній ОС. #os #resource

Ресурсом називають можливість, що забезпечується компонентами обчислювальної системи, що витрачається (займана) в процесі її роботи.

6. Перерахувати стратегії, що застосовуються в алгоритмах оптимізації. #strategy #algo #optimiz

Генетичних алгоритмів, оціночних функцій, прямого пошуку, Апеалінг метод, метод пошуку максимального паросполучення у зваженому графі, модифікований угорський, метод гілок та границь, модифікований метод для RealTimeOS, евристичний, складання розкладу, метод виключного планування.

7. Чим алгоритм FВп гірше алгоритму «Корбато»? #fbn #korbato #corbato

Перевага Корбато над іншими(списано з теорії до лабораторної): Корбато дозволяє скоротити кількість системних перемикань за рахунок того, що програмам, що вимагають більшого часу рішення, надаватимуться чималі кванти часу вже при першому занятті ними ресурсу. Можлива ще така відповідь(привильність невідома):

Алгоритм Корбато краще алгоритму FВп, т.к. потенційно володіє більшою продуктивністю тому розподіляє завдання по чергах не з їх пріоритетів, а за ознакою - абсолютна довжина коду програми (чим менше код тим вище пріоритет) (В систему додано аналізатор, який відразу розміщує заявки в свою чергу, відповідно середній Час очікування зменшується)

8. Сформулюйте принцип оптимальності Белмана. #belman #optim

Евристичні (засновані на досвіді людей), Покрокове конструювання - поділ всього процесу рішення на певну кількість кроків і знаходження оптимального або квазіоптимального рішення. ЦЕ - відповідає принципу оптимальності Белмана.

9. Назвати стратегії вирішення задачі розподілу завдань по ресурсах з пересилками.

10. Як визначити зону знаходження оптимального розкладу з урахуванням пересилань?

11. Визначити умови переходу зі стану P2 в стан P1. #p2 #p1 #pereh #state

Переходить з P2 в P1 якщо процеси знаходяться в одній черзі

12. Навіщо потрібно виконувати структурний аналіз вихідного графа? Чи можна його робити в динамічному режимі? #analys #struct #graph

Для подальшої оптимізації

13. Дати характеристику однорідної системи. #odnorodn #sys

Система однорідна, коли задачі в неї поступають однаково.

14. Математична постановка задачі динамічного планування. #math #task #dynamic

Задача комбінаторики по розміщенню задач в просторі та у площині.

15. Чому в розподілених системах переважніше використовувати крупнозернисте планування? #plan #zern #distrib

Крупнозернисте планування – це коли час виконання вузла > часу пересилки.