БІЛЕТ №9

1. Які основні характеристики розподілених систем обробки інформації ?

Розподілена система — 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 одноманітно 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 полагоджені 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.

Наступление события, освобождения ресурса, завершение ввода/вывода.

3. Коли застосовують евристичні алгоритми?

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. 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.

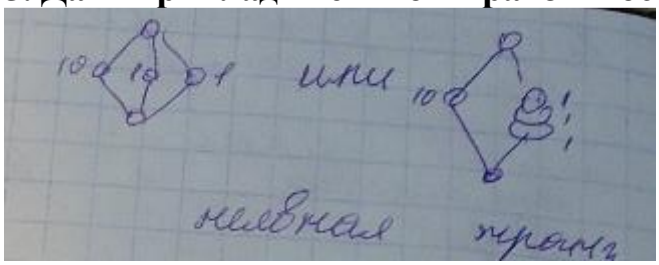
Дві фундаментальні цілі в інформатиці - знаходження алгоритмів з вірогідно найкращим часом виконання та з хорошою або оптимальною якістю. Евристичний алгоритм відмовляється від однієї або обох цих цілей; наприклад, він зазвичай знаходить дуже хороше рішення, але немає доказів, що рішення насправді не 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.

4Методи підвищення ефективності використання ресурсів обчислювальної системи.

5. Дати приклад «неявної» транзитності.



6. Що називається ядром супервізора, які його функції?

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

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?

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?

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?

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

14. Закон Амдала.

Зако́н 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 α от общего объёма вычислений может быть получена только последовательными расчётами, а, соответственно, доля $1-\alpha$может быть распараллелена идеально (то есть время вычисления будет обратно пропорционально числу задействованных узлов p). Тогда ускорение, которое может быть получено на вычислительной системе из p процессоров, по сравнению с однопроцессорным решением не будет превышать величины

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

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

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 u/ресурсов. (NP-полная – временна сложность - експонента)