Operating System | Deadlock detection algorithm
Operating System | Deadlock detection in Distributed systems
Techniques used in centralized approach of deadlock detection in distributed systems
How can one become good at Data structures and Algorithms easily?
Quick Sort vs Merge Sort
Minimum number of items to be delivered
Find maximum in a stack in O(1) time and O(1) extra space
Cracking Technical Interviews
Reach the numbers by making jumps of two given lengths
Print the nodes of binary tree as they become the leaf node
Find the number of distinct islands in a 2D matrix
Samsung Semiconductor Institute of Research(SSIR Software) intern/FTE | Set-3
Rearrange Odd and Even values in Alternate Fashion in Ascending

Operating System | Banker's Algorithm : Print all the safe state (or safe sequences)

Prerequisite – Resource Allocation Graph (RAG), Banker's Algorithm, Program for Banker's Algorithm

Banker's Algorithm is a resource allocation and deadlock avoidance algorithm. This algorithm test for safety simulating the allocation for predetermined maximum possible amounts of all resources, then makes an "s-state" check to test for possible activities, before deciding whether allocation should be allowed to continue.

In simple terms, it checks if allocation of any resource will lead to deadlock or not, OR is it safe to allocate a resource to a process and if not then resource is not allocated to that process. Determining a safe sequence(even if there is only 1) will assure that system will not go into deadlock.

Banker's algorithm is generally used to find if a safe sequence exist or not. But here we will determine the total number of safe sequences and print all safe sequences.

The data structure used are:

- Available vector
- Max Matrix
- Allocation Matrix
- Need Matrix

Example:

Input :

Total Resources	R1 R2 R3 10 5 7

Most popular in Algorithms
Iterative Letter Combinations of a Phone Number
Maximize the value of A by replacing some of its digits with digits of B
Array Manipulation and Sum
Length of the longest alternating subarray

Order

Shannon-Fano Algorithm for Data Compression

Sum of similarities of string with all of its suffixes

Maximize the total profit of all the persons

Lower and Upper Bound Theory

Dijkstra's shortest path with minimum edges

Sort the array of strings according to alphabetical order defined by another string

Smallest power of 2 which is greater than or equal to sum of array elements

Check if it is possible to reach a number by making jumps of two given length

Program for SSTF disk scheduling algorithm

Code Optimization Technique (logical AND and logical OR)

Dividing a Large file into Separate Modules in C/C++, Java and Python

Program to print the Zigzag pattern

Number of array elements derivable from D after performing certain operations

Process	Allocation			Max		
	R1	R2	R3	R1	R2	R3
P1	0	1	0	7	5	3
P2	2	0	0	3	2	2
P3	3	0	2	9	0	2
P4	2	1	1	2	2	2

```
Output : Safe sequences are:
P2--> P4--> P1--> P3
P2--> P4--> P3--> P1
P4--> P2--> P1--> P3
P4--> P2--> P3--> P1

There are total 4 safe-sequences
```

Explanation –
Total resources are R1 = 10, R2 = 5, R3 = 7 and allocated resources are R1 = (0+2+3+2 =) 7, R2 = (1+0+0+1 =) 2, R3 = (0+0+2+1 =) 3. Therefore, remaining resources are R1 = (10 – 7 =) 3, R2 = (5 – 2 =) 3, R3 = (7 – 3 =) 4.

Remaining available = Total resources – allocated resources
and
Remaining need = max – allocated

Process	Max			Allocation			Available			Needed		
	R1	R2	R3	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	7	5	3	0	1	0	3	3	4	7	4	3
P2	3	2	2	2	0	0				1	2	2
P3	9	0	2	3	0	2				6	0	0
P4	2	2	2	2	1	1				0	1	1
				7	2	3						

So, we can start from either P2 or P4. We can not satisfy remaining need from available resources of either P1 or P3 in first or second attempt step of Banker's algorithm. There are only four possible safe sequences. These are :
P2–> P4–> P1–> P3
P2–> P4–> P3–> P1

Remove characters from a numeric string such that string becomes divisible by 8

Most visited in GATE CS

How to prepare for GATE CSE - 2019

GATE CSE Test Series - 2019 | Sudo GATE

Functions of Operating System

Preemptive and Non-Preemptive Scheduling

Memory Hierarchy Design and its Characteristics

Largest perfect square number in an Array

How to write a Pseudo Code?

Color N boxes using M colors such that K boxes have different color from the box on its left

Smallest Pair Sum in an array

P4–> P2–> P1–> P3
P4–> P2–> P3–> P1

Recommended: Please try your approach on *{IDE}* first, before moving on to the solution.

Code –

```
// CPP Program to Print all possible safe sequences using banker's algorit
#include <iostream>
#include <string.h>
#include <vector>
// total number of process
#define P 4
// total number of resources
#define R 3

// total safe-sequences
int total = 0;

using namespace std;

// function to check if process
// can be allocated or not
bool is_available(int process_id, int allocated[][R],
                  int max[][R], int need[][R], int available[])
{

    bool flag = true;

    // check if all the available resources
    // are less greater than need of process
    for (int i = 0; i < R; i++) {

        if (need[process_id][i] > available[i])
            flag = false;
    }

    return flag;
}

// Print all the safe-sequences
void safe_sequence(bool marked[], int allocated[][R], int max[][R],
                   int need[][R], int available[], vector<int> safe)
{

    for (int i = 0; i < P; i++) {

        // check if it is not marked
        // already and can be allocated
        if (!marked[i] && is_available(i, allocated, max, need, available))

            // mark the process
```


Related Articles

Difference between Spooling and Buffering

XOR of XORs of all sub-matrices

Kruskal's Algorithm (Simple Implementation for Adjacency Matrix)

Implementing Directory Management using Shell Script

Minimum steps to reach any of the boundary edges of a matrix | Set 1

Minimum operations of given type to make all elements of a matrix equal

Find probability of selecting element from kth column after N iterations

```
            marked[i] = true;

            // increase the available
            // by deallocating from process i
            for (int j = 0; j < R; j++)
                available[j] += allocated[i][j];

            safe.push_back(i);
            // find safe sequence by taking process i
            safe_sequence(marked, allocated, max, need, available, safe);
            safe.pop_back();

            // unmark the process
            marked[i] = false;

            // decrease the available
            for (int j = 0; j < R; j++)
                available[j] -= allocated[i][j];
        }
    }

    // if a safe-sequence is found, display it
    if (safe.size() == P) {

        total++;
        for (int i = 0; i < P; i++) {

            cout << "P" << safe[i] + 1;
            if (i != (P - 1))
                cout << "--> ";
        }

        cout << endl;
    }
}

// Driver Code
int main()
{

    // allocated matrix of size P*R
    int allocated[P][R] = { { 0, 1, 0 },
                            { 2, 0, 0 },
                            { 3, 0, 2 },
                            { 2, 1, 1 } };

    // max matrix of size P*R
    int max[P][R] = { { 7, 5, 3 },
                      { 3, 2, 2 },
                      { 9, 0, 2 },
                      { 2, 2, 2 } };

    // Initial total resources
    int resources[R] = { 10, 5, 7 };

    // available vector of size R
    int available[R];
```

Count the total number of squares that can be visited by Bishop in one move

Memory Hierarchy Design and its Characteristics

Number of submatrices with all 1s

Fixed (or static) Partitioning in Operating System

Print numbers in matrix diagonal pattern

Matrix Chain Multiplication (A O(N^2) Solution)

Paytm Interview experience for FTE (On-Campus)

Print all paths from top left to bottom right in a matrix with four moves allowed

```
        for (int i = 0; i < R; i++) {

            int sum = 0;
            for (int j = 0; j < P; j++)
                sum += allocated[j][i];

            available[i] = resources[i] - sum;
        }

        // safe vector for displaying a safe-sequence
        vector<int> safe;

        // marked of size P for marking allocated process
        bool marked[P];
        memset(marked, false, sizeof(marked));

        // need matrix of size P*R
        int need[P][R];
        for (int i = 0; i < P; i++)
            for (int j = 0; j < R; j++)
                need[i][j] = max[i][j] - allocated[i][j];

        cout << "Safe sequences are:" << endl;
        safe_sequence(marked, allocated, max, need, available, safe);

        cout << "\nThere are total " << total << " safe-sequences" << endl;
        return 0;
    }
```

Output:

```
Safe sequences are:
P2--> P4--> P1--> P3
P2--> P4--> P3--> P1
P4--> P2--> P1--> P3
P4--> P2--> P3--> P1

There are total 4 safe-sequences
```

Recommended Posts:

Print all safe primes below N

Find safe cells in a matrix

Find shortest safe route in a path with landmines

Operating System | Banker's Algorithm

Operating System | Dekker's algorithm
Operating System | Bakery Algorithm
Operating System | Deadlock detection algorithm
Operating System | Program for Next Fit algorithm in Memory Management
Operating System | Peterson's Algorithm (Using processes and shared memory)
Operating System | Buddy System - Memory allocation technique
Operating System | Starvation and Aging in Operating Systems
Operating System | Semaphores in operating system
Operating System | Introduction of Operating System - Set 1
Operating System | Requirements of memory management system
Operating System | Types of Operating Systems

shubham_rana_77
Check out this Author's contributed articles.

If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please Improve this article if you find anything incorrect by clicking on the "Improve Article" button below.

Article Tags : Algorithms GATE CS Matrix Operating Systems

Practice Tags : Operating Systems Matrix Algorithms

Be the First to upvote.

To-do Done

4

Based on 1 vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Previous
Mathematics | Graph Theory Basics – Set 1

Next
GATE CS 2018 Mock Tests

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments Share this post!