Extended Banker’s Algorithm for Effective Human Resource Allocation in Project Management

Harshal Gurjar  
V.J.T.I. College  
Mumbai-19  
harshalkgurjar@gmail.com

Rakesh Waghulde  
V.J.T.I. College  
Mumbai-19  
waghulderakesh@gmail.com

Abstract

Human resource allocation and management is an important aspect in efficient management of any project. Deadlock-free management of resources is essential for organized project management. Banker’s algorithm, which was primarily developed for computers operating system, with certain modifications, can be used for the effective management of human resources for successful completion of the projects. Based on the availability and requirement of the different human resources and their expertise, the algorithms analyses the possibility of success of the projects. The algorithm efficiently allocates the human resource for each processes of the business according to the various specifications. Further, this algorithm also recognizes the number of surplus resources or shortage in resources, if any. This method has a potential to provide deadlock-free operation as well as guarantees optimized resource management.

General Terms

Algorithms, Banker’s algorithm, Resource management, Deadlock

Keywords

Safe-state, HRM-human resource management, Deadlock, Deadlock avoidance, Resources (Resources denotes human resources)

Introduction

Human resource management is relatively new approach for managing people in an organization. People are considered key resources in this approach. It is concerned with the synchronization and allocations of human resource of different expertise to the available projects. It deals with the scheduling and assigning of human resources to different projects based on the availability and requirement of the resource.

Deadlock-free allocation of resource is one of the important aspects of human resource management. In general, deadlock is a situation in which each process is waiting for another process to release the required resources, which in turn leads to blocking of all the processes in the system. The imprudent allocation of resources and inadequate buffer space can lead to deadlock. Human Resource management, therefore, must incorporate some strategy for handling deadlock; otherwise smooth operation of project cannot be guaranteed.

Further, imprudent resource allocation increases the number of resources required for certain projects, which in turn increases the cost of the projects. Banker’s algorithm provides solution to this problem, by synchronizing the processes and optimizing the allocated resources.

Banker’s algorithm for HRM

Banker’s algorithm was developed by Edsger Dijkstra. The Banker’s Algorithm derives its name from the fact that this algorithm could be used in a banking system to ensure that the bank does not run out of resources, because the bank would never allocate its money in such a way that it can no longer satisfy the needs of all its customers. By using the Banker’s algorithm, the bank ensures that when customers request money the bank never leaves a safe state. If the customer's request does not cause the bank to leave a safe state, the cash will be allocated, otherwise the customer must wait until some other customer deposits enough.

Banker’s algorithm is most commonly used to manage various resources by operating system. Banker’s algorithm can be used for allocation of different resources to different process, while preventing the deadlock of the processes. Banker’s algorithm was initially developed for operating system to manage the allocation of different resources to different process.

Firstly, banker’s algorithm simulates the allocation of maximum claimed resources by any process and then checks for the possible deadlock condition. If the system is deadlock free, then it is considered as a safe-state. Further, if there is possible occurrence of deadlock, banker’s algorithm determines the minimum number of resources to be allocated to each process to make it deadlock-free.
Banker’s algorithm needs 3 inputs:
1. Maximum number of each category of human resources each process can request [Max_Request]
2. Number of each category of human resources already allocated to each process [Allocated_Res]
3. Number of each category of human resources available in the system [Avail_Res]

Resources can be allocated to process only if it satisfies the following conditions:
1. Request <= Max_Request
2. Request <= Avail_Res

Basic data structures to be maintained to implement the Banker’s Algorithm:
Let n be the number of processes in the system and m be the number of resource types. Then we need the following data structures:

- **Avail_Res**: A vector of length m indicates the number of available resources of each type. If Available[j] = k, there are k instances of resource type Rj available.
- **Max_Request**: An n×m matrix defines the maximum demand of each process. If Max[i,j] = k, then Pi may request at most k instances of resource type Rj.
- **Allocation_Res**: An n×m matrix defines the number of resources of each type currently allocated to each process. If Allocation[i,j] = k, then process Pi is currently allocated k instances of resource type Rj.
- **Need**: An n×m matrix indicates the remaining resource need of each process. If Need[i,j] = k, then Pi may need k more instances of resource type Rj to complete the task.

Also, Need = Max_Request - Avail_Res

The working, input and output of the algorithm will be clearer with the following tables:

**Input:**

**Claimed matrix:**

<table>
<thead>
<tr>
<th></th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>26</td>
<td>21</td>
<td>6</td>
</tr>
<tr>
<td>P2</td>
<td>12</td>
<td>56</td>
<td>2</td>
</tr>
<tr>
<td>P3</td>
<td>12</td>
<td>54</td>
<td>89</td>
</tr>
</tbody>
</table>

4. Algorithm prevents the deadlock of the resources.

**Allocated matrix:**

<table>
<thead>
<tr>
<th></th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>P2</td>
<td>67</td>
<td>45</td>
<td>1</td>
</tr>
<tr>
<td>P3</td>
<td>12</td>
<td>9</td>
<td>4</td>
</tr>
</tbody>
</table>

**Available matrix:**

<table>
<thead>
<tr>
<th></th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

**Output:**

- Hire 11 resources of R1.
- Hire 5 resources of R2.
- Project P1 is completed.
- Project P2 is completed.
- Hire 67 resources of R3.
- Project P3 is completed.

**Future scope:**

1. Current system suggests the number of employees/resources to hire. But, in most cases, all the resources in assigned to the process are not utilized throughout the execution of project. So, many different project projects can share the resources. Thus reducing the number of resources required to the company.
2. Web portal can be created for an industry which would contain the available resources in the industry. Further considering the available resources, each project lead can propose their requirement of resources.

**Advantages:**

1. Algorithm optimizes the resources required for each project.
2. Optimized resource allocation reduces the cost of the project.
3. Algorithm works in a real time.
Conclusion:
In this paper we proposed the extended banker’s algorithm for human resource allocation in project management. The algorithm successfully prevents the deadlock in resource allocation. Also algorithm provides the optimized utilization of resources. Thus reducing the number of resources required for particular project.

REFERENCES

3. “Operating system principles” by Peter Galvin
4. "Deadlock prevention algorithm " wikipedia: