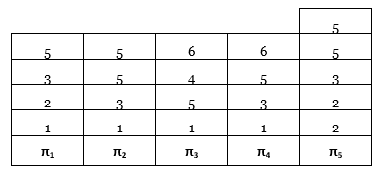
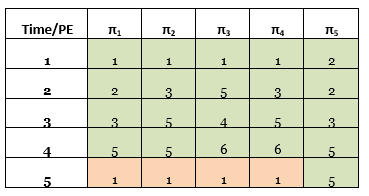
**Discussions:**

1. **Spatial and Temporal Scheduling of Jobs:**

For the given distributed computing system, we first try to allocate the VPs of the Jobs (**J1 - J6)** in such a way that the resulting **Scheduling Matrix** from that **Allocation Table** results not only in a legal schedule but also but also tries to achieve the **0%** idling ratio.

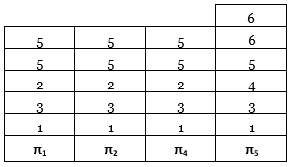
The **Allocation Table** we provide alongside, has allocated a certain number of VPs of each of the Jobs (**J1 - J6)** to each Processor in a way to achieve the best Temporal Schedule. For example, the 3rd Processor (**π3**) is allocated 1 VP each of Jobs (**J1, J5, J4** and **J6**) respectively. The same logic extends for all other Processors (**π1**, **π2, π4** and **π5**) and explain the Allocation Table.

To derive a legal temporal scheduling, we invert the allocation table to get a **Scheduling Matrix**. The matrix thus obtained would look as below. In the matrix, each **green grid** corresponds to a particular Job’s **VP** as assigned to a Processor (**πx**) given by the Allocation Table.

To prove that it is legal, we consider each time slab one at a time. For example, in **Time Slice 1**, the Processor **π5** advances the **VP-1** of Job-2 by one time slice. In **Time Slice 2**, the Processor **π1** and **π5** advances the other **VP-2** and **VP-3** of the same Job 2 by one time slice. Since **π5** has been allotted two VPs of Job 2, incrementation in the second time slice, does not break the condition of **Legal Scheduling** as it does not increment the previously incrementing VP. At the end of the 5th Time Slice, all the VPs of all the Jobs have been incremented exactly by 1. Here, we can say that one period is complete.

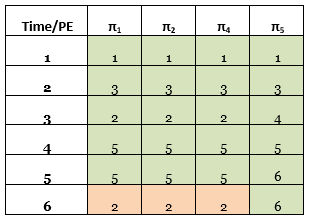
To make up for the idling time of the processors (**π1** to **π4**) in the 5th time slice, a new run of Job-1 has been assigned in the **orange grids**. To calculate the **idling ratio**, we see that out of a total 25 slots all of them are occupied in the given period. Hence, for the above-mentioned scheduling matrix, the

1. **Handling failure of Processor π3:**
   1. **New Allocation Table:**



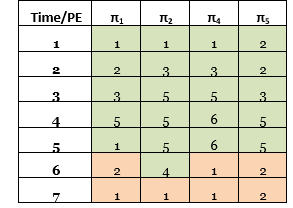
There are two ways to handle the failure of a processor in a distributed computing system. First, we consider creating a **new Allocation Table** for all the jobs after removing the given faulty Processor **π3** from the system. The table given alongside is a new reallocation of the same VPs of all the Jobs trying to minimize the **Idling Ratio**.

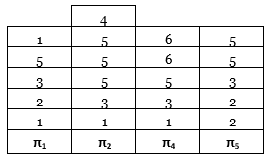
While we ensure that the allocation thus done, can still be used to create a **legal Scheduling Matrix**, we further try to minimize the Idling Ratio of our new matrix. The jobs have been redistributed without considering the overhead costs of “migrating VPs” but only taking the best idling ratio and reduced period time cycle into consideration.

 The resultant **Scheduling Matrix** is the one given alongside. It follows the same convention where each **green grid** corresponds to a particular Job’s VP allocated for that period. The **orange grids** indicate the next iteration of Job-2 allocated to that period to reduce the **Idling Ratio** to **0%** as there are no empty slots.

Although this matrix keeps the Idling Ratio to 0%, but it requires a lot of **VP Migration**. Hence, in the next approach, we migrate only the VPs assigned to the failed Processor **π3** to reduce VP Migration but still keep the Idling Ratio as high as possible.

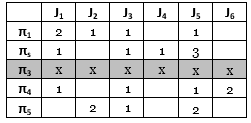
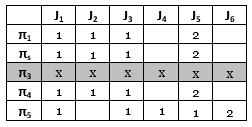
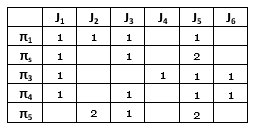
* 1. **Migrating Only Processor 3’s VPs:**

The new Allocation Table and the corresponding Scheduling Matrix obtained by migrating only the VPs of the failed Processor **π3** is provided below.



Although the period of the resultant Scheduling Matrix increases by 1, it still manages to achieve a 0% Idling Ratio by allocating the empty time slots to the next iteration of Jobs J1 and J2.

1. **Cost of Migration:**

The units of each Processor in the original allotment, after new table allocation and after completing migration of only faulty units of **π3** ­are shown as follows:

1. Initial Allocation Distribution (2a) New Allocation Table Distribution (2b) Distribution after migrating π3­’s VPs

From the table, we can confirm that the number of units migrated from the **Initial Distribution** **(1)** to the **New Allocation Table (2a)** Distribution is: 10 units.

The migration from the first table to the third table after distributing the **VPs of π3** is always **4 units** as only the units that were on Processor **π3** were distributed to the other Processors.

Thus, we can conclude that using this particular allocation arrangement to begin with, the cost of migrating VPs from the faulty processor to the other processors are less than the cost of reallocating all units using a new allocation table.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **J1** | **J2** | **J3** | **J4** | **J5** | **J6** |
| **π1** | 1 | 1 | 1 |  | 1 |  |
| **πs** | 1 |  | 1 |  | 2 |  |
| **π3** | 1 |  |  | 1 | 1 | 1 |
| **π4** | 1 |  | 1 |  | 1 | 1 |
| **π5** |  | 2 | 1 |  | 2 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **J1** | **J2** | **J3** | **J4** | **J5** | **J6** |
| **π1** | 1 | 1 | 1 |  | 2 |  |
| **πs** | 1 | 1 | 1 |  | 2 |  |
| **π3** | x | x | x | x | x | x |
| **π4** | 1 | 1 | 1 |  | 2 |  |
| **π5** | 1 |  | 1 | 1 | 1 | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **J1** | **J2** | **J3** | **J4** | **J5** | **J6** |
| **π1** | 2 | 1 | 1 |  | 1 |  |
| **πs** | 1 |  | 1 | 1 | 3 |  |
| **π3** | x | x | x | x | x | x |
| **π4** | 1 |  | 1 |  | 1 | 2 |
| **π5** |  | 2 | 1 |  | 2 |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | 4 |  |  |
| 1 | 5 | 6 | 5 |
| 5 | 5 | 6 | 5 |
| 3 | 5 | 5 | 3 |
| 2 | 3 | 3 | 2 |
| 1 | 1 | 1 | 2 |
| **π1** | **π2** | **π4** | **π5** |

1. **Temporal Schedule with Minimum Idling Ratio:**

Use this table. :P

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Time/PE** | **π1** | **π2** | **π4** | **π5** |
| **1** | 1 | 1 | 1 | 2 |
| **2** | 2 | 3 | 3 | 2 |
| **3** | 3 | 5 | 5 | 3 |
| **4** | 5 | 5 | 6 | 5 |
| **5** | 1 | 5 | 6 | 5 |
| **6** | 2 | 4 | 1 | 2 |
| **7** | 1 | 1 | 1 | 2 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| P 1 | 2 VP of J1 | 3 VP of J2 |  |
| P2 | 1 j1, 1 j3, 3 j5 |  |  |
| P3 |  |  |  |
| P4 |  |  |  |
| P5 |  |  |  |
|  |  |  |  |

|  |  |
| --- | --- |
|  |  |
| 1 | 4 |  |
| 1 | 5 | 6 | 5 |
| 5 | 5 | 6 | 5 |
| 3 | 5 | 5 | 3 |
| 2 | 3 | 3 | 2 |
| 1 | 1 | 1 | 2 |
| **π1** | **π2** | **π4** | **π5** |