**Operating Systems**

**Lab-6: Implementing Contiguous Memory Allocation Schemes**

Run the .cpp file to start contiguous memory allocation simulation with a memory block of 1024 kb.

**Input:**

After running the code, it takes two values.

First value (job size),

* 1 (Any sized jobs): Each job can be of any random size between 1 and 1024 kb.
* 2 (Small jobs): Each job can be of any random size between 1 and 100 kb.
* 3 (Large jobs): Each job can be of any random size between 500 and 1000 kb.

Second value (memory allocation polices),

* 1 (best fit)
* 2 (worst fit)
* 3 (first fit)

It runs the simulation for 2000, after 1000 simulations it starts calculating the following values and prints them in output.

**Output:**

● The average fragmentation (as a fraction of total memory)

● The average size of a hole

● The average number of holes that must be examined to satisfy a single new partition

request

● The highest fragmentation measured in any time step (as a fraction of total memory)

● The lowest fragmentation measured in any time step (as a fraction of total memory)

● The average number of jobs in memory

● The highest number of jobs ever in memory

● The lowest number of jobs ever in memory

● The average size of a job

● The average number of holes

● The highest number of holes recorded

● The lowest number of holes recorded

● The largest number of partitions created in a row without any intervening evictions

● The largest number of evictions required in satisfying a single new partition request

**Implementation:**

The contiguous memory is simulated using an array of size 1024 indexed from 1 to 1024. Each unit contains either 0 or 1. 1 indicating that the index is in use, 0 indicating that the index is not in use (is free).

If a job needs to be placed in memory, we set the range of values in array to 1.

If a job needs to be removed from memory, we set the range of values in array to 0.

A map is used to store where a job is allocated. The job\_id is stored as the key and the start and end index(inclusive) of contiguous memory array is stored as a pair of those indices as the value in the map if that job\_id exists in memory.

Another map is used to store job\_id as key along with its size as value.

A random number generator is used to generate the size of the specified job profile range.

First fit function traverses the memory array and claims the first available memory with space more than or equal to its size and updates the maps and contiguous memory array with the changes.

Best fit function traverses the whole memory array and always search for the smallest hole/partition such that the hole size is more than or equal to its size, and then the process is placed in that hole/partition and updates the maps and contiguous memory array with the changes.

Worst fit function traverses the whole memory array and always search for the largest hole/partition, and then the process is placed in that hole/partition and updates the maps and contiguous memory array with the changes.

If a job is unable to be placed in the memory array, then we call the remove\_job function.

Remove job function removes a randomly selected job already present in the memory (frees up space) and updates the maps and contiguous memory array with the changes.