Consider a set of **J** jobs (arrived at time 0) that need to be processed by a cloud consisting of **N** physical machines (i.e., nodes) that are homogeneous. Each job j has a deadline $\mathbf{d_j}$ and is required to access a set $\mathbf{C_j}$ of equal-sized chunks, we can assume each job j has $\mathbf{|C_j|}$ number of tasks accessing one data chunk each and can be run in parallel on the same machine or different machine.

The chunks are stored in a distributed file system on the cloud. Each node is capable of hosting up to **B** data chunks and is equipped with **S** virtual machines which implies each node is able to simultaneously process S jobs. Let $\mathbf{C} = \cup \mathbf{C_j}$ be the set of all data chunks available at the central storage server, before processing the request we need to bring the data chunk to the physical machine.

Replication of data chunks in different machines is allowed and data chunk needs to be placed only once at the beginning (before any job starts execution, once any one job starts the execution, we are not allowed to change the data placement) and during run time we can not place/replace/replicate the data chunk. The time for each job j to process a required data chunk is unit time and it is the same for all the jobs.

Completing a job j before the deadline d_j is equivalent to processing all the required chunks c of C_j before the deadline. Only one VM can access a data chunk in a given time slot of the same physical machine. The problem aims to minimise the **total number of active nodes ($\mathbf{N_a}$)** to process all the jobs. The active node means the node stores at least one data chunk and is processed by at least one job. The number of active machines is always lower than the total number of machines which is $\mathbf{N_a < N}$.