***Abstract*—Clusters and supercomputers make use of workload schedulers such as the Slurm Workload Manager or OAR to allocate computing jobs onto nodes. These schedulers usually aim at a good trade-off between increasing resource utilization and user satisfaction (decreasing job waiting time). However, these schedulers are typically unaware of jobs sharing large input files: in data-intensive scenarios, tens to hundreds of jobs dedicated to the study of the same multi-GB input file may be successively submitted. Running each of these jobs first requires to load the input file, leading to a large data transfer overhead. Users could manually group some of these tasks into larger jobs to reduce data transfers, but this would result in less granular units that are more difficult to schedule by the resource manager, and would thus result in a larger delay as well. We study how to design a data-aware job scheduler that is able to keep large input files on the computing nodes, provided this does not impact the memory needs of other jobs, and can use previously loaded files to limit data transfers in order to reduce the waiting times of jobs.**

**We present three schedulers capable of distributing the load between the computing nodes as well as being aware of what the memory of each node contains in order to re-use an input file already loaded in the memory of some node as many times as possible.**

**We report simulations performed using real cluster usage traces. Our approach is compared to currently used schedulers in batch systems. The results show that keeping data locally, in memory, between successive jobs and using data locality information to schedule jobs allows a reduction in job waiting time and a drastic decrease in the amount of data transfers.**

***Index Terms*—Job input sharing, Data-aware, Job scheduling, High Performance Data Analytics**