The NorduGrid Brokering Algorithm

1 Finding Resources

The search for available resources can either use the Grid Information Index Service (GIIS) to find available clusters, or use an explicit list of clusters. By default the GIIS method is used unless the user uses the -c or -C option with ngsub or the -k or -K option with ngresub.

When the GIIS method is used a list of top level GIIS servers must be available. If the user uses the -g or -G option with ng(re)sub, this list is compiled from these options, otherwise the first available file in the following list is used:

- ${HOME}/.nggiislist
- ${NORDUGRID_LOCATION}/etc/giislist
- /etc/giislist

When a standard installation of the NorduGrid toolkit is used, a default list of GIIS servers is installed in ${NORDUGRID_LOCATION}/etc/giislist.

Each GIIS server is queried about its GIIS registrations. If a registration from another GIIS server is found it is added to the list of GIIS servers, unless it isn't already there. If registration from a cluster is found it is added to the list of clusters, unless it isn't already there. When all GIIS servers (the ones in the original list as well as the ones found during the search) have been queried the list of clusters is completed.

From the list of clusters, either the one obtained from the GIIS search, or the implicit list compiled from the command line options, any cluster rejected by using a -c or -C option to ngsub (or a -k or -K option to ngresub) with an argument starting with a minus sign are removed.

The Grid Resource Information Service (GRIS) servers on the found clusters are then queried about the characteristics and current state of the clusters and the queues available on them. Information about jobs not yet queued in the Local Resource Management System (LRMS), i.e. jobs in ACCEPTED and PREPARING states are also retrieved from the GRIS server, and the amount of free CPU's for each queue is adjusted to make room for these jobs.

2 Matching Resources

For each job that should be submitted the queues found on the available clusters are stepped through to find possible submission targets. Queues that report a different status than "active" are rejected as are queues where the user is not authorized to submit jobs, queues that have reached their queueing limit and queues that don't have any CPU's.

The characteristics of the cluster and the queue are then compared to the XRSL description of the job that is about to be submitted. The following attributes in the XRSL are considered: "cluster", "queue", "count", "cputime",

"gridtime", "memory", "architecture", "middleware", "runtimeenvironment", "opsys" and "nodeaccess". If any of these attributes can't be matched the queue is rejected.

At this point, the amount of needed disk space is calculated. The sizes of all the input files are obtained by querying the replica catalogue (RC), replica location service (RLS), gsiftp and http servers as specified in the XRSL description. Also the sizes of local input files are evaluated. The results of these queries are kept so that when more than one job is submitted using the same ng(re)sub command and these jobs use partly the same input files, the size of the common files don't have to be queried for more than once.

The input files are grouped into the following classes: REMOTE, LOCAL, REMOTENOCACHE, LOCALNOCACHE, CACHED and NOLOCATION, depending on where they are located w.r.t. the cluster that is being considered as a target, and any "local" and "cache" options given in the XRSL. If any of the input files end up in the NOLOCATION class (e.g. an input file that has a "local" option set, but is not available locally on the cluster) the queue is rejected. Local files that will be uploaded from the submitting computer are grouped in the REMOTENOCACHE class.

The sizes of the files are then added into two sums, one for those that will be downloaded to the cache (REMOTE, LOCAL) and one for those that will end up in the job's session directory (REMOTENOCACHE, LOCALNOCACHE). The size of the job's output files given by the "disk" attribute in the XRSL is added to the latter sum. These sums are then compared to the available space in the cache and the user's free disk space. If there is no cache on the cluster being considered, the sum of the sums is compared to the user's free disk space. If the sizes are to big the queue is rejected.

3 Choosing the Submission Target

After having considered all the available queues, the remaining queues are the possible targets. At this point two things can happen. Either there exists at least one possible target where the amount of free CPU's available for the user is greater than or equal to the number of CPU's specified in the XRSL, or there are no such targets.

If there are targets with free CPU's available, these are ranked according to the total size of the files that have to be transferred from a remote site (i.e. the files in the REMOTE and REMOTENOCACHE classes). The target with the smallest size is chosen. If there is more than one such cluster these in turn are ranked according to the total size of the files that must be downloaded from a local site (i.e. the LOCAL and LOCALNOCACHE classes). If there is still more than one cluster with the smallest total size, one of these is chosen at random with a weight given to each queue proportional to the number of free CPU's available for the user in that queue.

If there are no targets with enough free CPU's available, the job is submitted to the target where the ratio between the number of queued jobs and the total number of CPU's is the smallest.

If the submission of the job to the selected queue fails the queue is rejected and a new target is chosen among the remaining possible targets using the same algorithm, until there are no more targets available.

If the submission is successful, the information about available CPU's and available disk space on the selected target is adjusted for the resources that will be used by the just submitted job. Input files that the job will download to the cache on the cluster will be tagged, so that if a subsequent job will use the same file it will be grouped in the CACHED class when the same cluster is considered for that job.

The submission then continues with matching resources for the next XRSL job description.