Fundamentals of Grid Computing

Alecu Felician, ASE Bucharest

A grid is a collection of machines that contribute any combination of resources as a whole. Basically, grid computing represents a new evolutionary level of distributed computing. It tries to create the illusion of a virtual single powerful computer instead of a large collection of individual systems connected together. These systems are sharing various resources like computing cycles, data storage capacity using unifying file systems over multiple machines, communications, software and licenses, special equipments and capacities.

The use of the grid is often born from a need for increased resources of some type. Grids can be built in all sizes, ranging from just a few machines in a department to groups of machines organized in hierarchy spanning the world. The simplest grid consists of just few machines, all of the same hardware architecture and same operating system, connected on a local network. Some people would call this a cluster implementation rather than a grid. The next step is to include heterogeneous machines but within the same organization. Such a grid is also referred to as an *intragrid*. Security becomes more important as more organizations are involved. Sensitive data in one department may need to be protected from access by jobs running for other departments. Dedicated grid machines may be added to increase the service quality. Over time, a grid may grow to cross organization boundaries and may be used for common interest projects. This is known as an *intergrid*.

The easiest way to use a grid is to remotely run an application on a different computer than the one on it is usually executed. If the computer that usually runs the job is busy, it can execute the application on another idle machine from the grid network. The remote machine must meet hardware, software and resource requirements of the application. Desktop machines from most organizations are underutilized because they are busy less then 5% of time. Grid computing is able to increase the resource usage efficiency because it could be obtained a better balance of resource utilization. If an application is grid enabled it could be moved to an idle computer from the grid whenever the host computer is busy.

If an application is written to use algorithms that can be divided into independent parts than each part could be executed on a different machine in the grid. This is why the grid computing offers a high potential for massive parallel CPU capacity. This huge computing power obtained by the use of the grid is driving a new evolution in various industries like financial modeling, oil exploration, bio-medical field and so on. Scalability is the measure of how efficiently the grid processors are used. A perfect scalable application will finish *n* times faster when it uses *n* times the number of processors. It is very hard to achieve the perfect scalability. The limits of scalability are called barriers. An example of such a barrier is the situation when an application can be split only into a limited number of independent parts. Not completely independent parts and communications between jobs are another barrier examples because they limit the scalability. Figure 1 shows how the application parts are executed concurrently in a grid network.

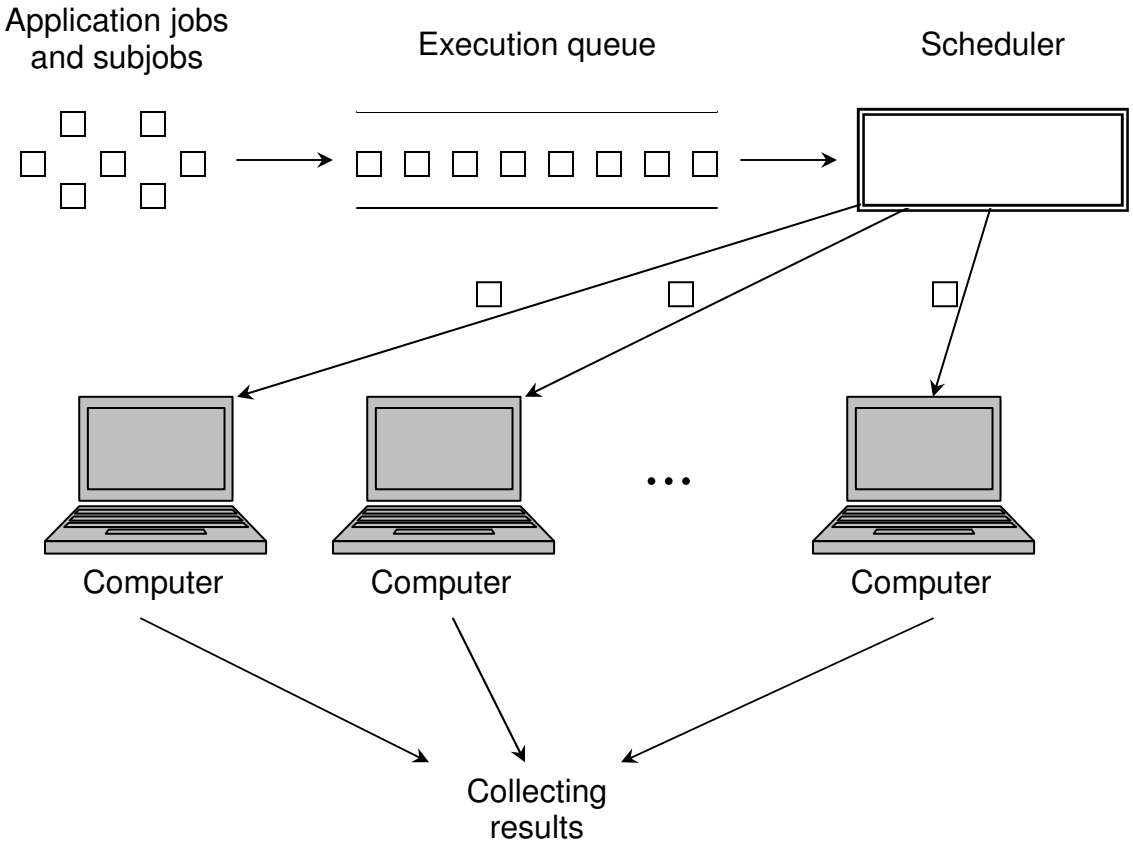

Figure 1 – Running an application in a grid network

Dependencies prevent jobs to be executed in parallel in all cases. Jobs may spawn additional subjobs. The result is a hierarchy of jobs and subjobs. The results of all jobs must be collected in order to obtain the final result of the application.

The grid network is considered like a single center and the service demands are scheduled on the individual computers from the grid. There is a single queue of jobs for all processors from the network. The total service rate of *n* processors from the grid can be significantly les then *n* times the rate of a single processor because of competition for software locks (such of those controlling access to the shared queue of jobs) and interference in accessing shared resources. On the other hand, the effective service rate of the grid network is not constant but depends on the number of jobs queued at center. Consider a four processor system. Ideally, if four or more jobs desire service at the center, all four processors can be kept busy, and the effective service rate of the center is its maximum rate. However, if less than four jobs are queued at the center, some of the machines will be idle, and so the effective service rate will de reduced correspondingly. Figure 2 graphs effective service rate as a function of the queue length for a four processor system. Service rates increase with queue length until all four systems are busy, after which increasing the number of jobs contending for the processors does not result in any increase in effective service rate. The dashed line illustrates the ideal growth in service rate and the other one represents the effect of contention.

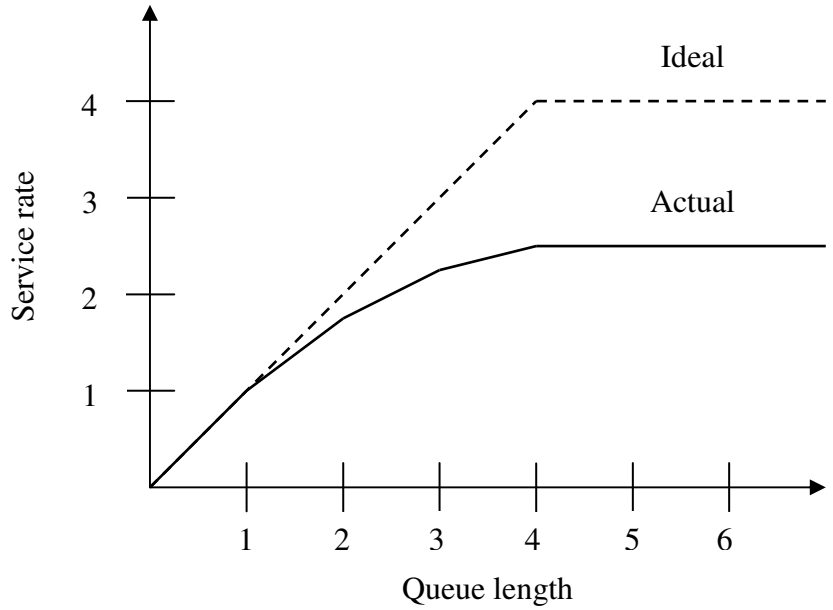

Figure 2 – Service Rate Function of a Four Processor Grid

Grid computing could be used also to run an application that needs to be executed many times on the computers from the grid network. In such a way the results could be obtained faster using the grid.

There are many factors to consider writing a grid enabled application. Application designers can use tools to write parallel grid applications. There are no tools for transforming applications to use the parallel capabilities of a grid. Some already existing applications could not be transformed to run in parallel on a grid. By scheduling jobs on underutilized machines the grid offers a resource balancing effect for grid enabled applications. The scheduler can migrate jobs to less busy parts of the grid to balance resource loads and absorb unexpected peaks of organization activity. The scheduler could schedule jobs to reduce the communications traffic or to minimize the distance of the communications.

The most common resource used in grid networks is computing cycles provided by the processors of the machines on the grid. Scalability is used to measure how efficiently the multiple processors on a grid are used. The second most common resource is data storage and it could be memory attached to a processor (very fast but volatile) or a permanent storage media like a hard disk. A large file can be stored on multiple machines using a unifying file system that provides a single uniform name space for grid storage. The users don't need to know the exact location of referenced data. The jobs that are using data have to be scheduled closer to the data, preferably on the node where the data resides. The grid file system has to implement synchronization mechanism in order to avoid contention when many users concurrently update shared data.

Another important resource of a grid is data communication capacity that is very important for sending jobs and their required data to points within the grid. Some jobs require a large amount of data to be processed that could not reside on the machine running the job. In such a case the bandwidth could be a critical resource that can limit utilization of the grid.

If we have software that is very expensive to be installed on every machine from organization, a grid could send the jobs that require this software to the machines on which it is installed. This approach saves significant expenses when the licensing fees are considerable.

The shared resources are usually accessed via an executing application. Applications may be broken down into any number of individual jobs, as illustrated in figure 1. In turn, jobs can be further broken down in subjobs. Independent jobs will be executed in parallel on different computers from the grid network. Some jobs could not be executed in parallel because they have specific dependencies that may prevent them from executing in parallel in all cases. Finally, the results of all jobs are collected and assembled to produce the ultimate answer for the application.

The scheduler is responsible for sending a job to a given machine to be executed. In the simplest of grid systems, the user selects a machine and executes a grid command that sends the job to that machine. More advanced grid systems include a job scheduler that automatically finds the most appropriate machine for a waiting to be executed task. Gird machines that are not dedicated to the grid are suspending grid tasks when local non-grid work has to be done. The dedicated machines are not preempted by outside work. Schedulers try to assign executing jobs to machines nearest to the data that these jobs require in order to decrease the network traffic and possibly reduce scalability limits. Optimal scheduling is still a difficult mathematics problem.

A few programmers may install a grid in their spare time. The planning become essential as the grid grows and the users will be more dependent on it. Security is much more important factor in planning and maintaining a grid than in conventional distributed computing. Any grid system has some management components used to keep track of the resources available to the grid. The scheduler uses this information in order to decide where grid jobs should be assigned. In the simplest case, the scheduler may assign jobs in a round-robin fashion. Some schedulers implement a priority system using several jobs queues, each with a different priority. When a grid machine become available, the job will be taken from the highest priority queue. Policies of various kinds are also implemented using schedulers.

Not every application is suitable for running in parallel on a grid. Some applications simply cannot be parallelized. For others, it can take a large amount of work to modify them to run concurrently.

Romanian companies are registered as taxpayers to the territorial tax office. The companies have the obligation to prepare and submit an annual balance sheet to the

territorial tax office, and also a balance sheet as of the half of the year. The balance sheet is an accounting document containing information about the company's activity during the year, by showing the situation of items such fixed assets, land, inventories, receivables, short-term debts, share capital, reserves, long-term liabilities at the end and beginning of the reporting period (i.e. year, half of year).

Billion of balance sheets are submitted every year and every 6 months to the territorial tax offices, both on a hard copy and on floppy disks. The territorial tax authorities collect all the data and introduce it in their informatics system. By aggregating information containing in the balance sheets submitted by companies, the authorities obtain useful information about the overall economy. Thus, the flow of fixed assets, land and share capital, where the flow is calculated as difference between the statement as of the end of the reporting period and the beginning of the reporting period, is aggregated on regions, on specific industries and finally on the entire economy and what is obtained is very important: the investment in share capital and in fixed assets during the year, in different regions, in different industries. Thus, the new investments are monitored and based on them the Government establishes its policy for the future in order to stimulate investments in certain poor regions, in certain industries, where without significant support, no economic growth will be realized.

Grid computing could be successfully used to consolidate these accounting data in a faster and more accurate manner by executing a data-parallel application in every processing node. The data space could be divided into multiple parts and every computer from the network will process such a part. Once a computer is idle, the scheduler assigns it a data set to be processed. After the computer finishes its work, the results will be send to the central processing node (who runs the scheduler too) and the computer will wait to receive another data set to process. The same set of instructions will be executed on different data sets. The intermediate results will be aggregated in order to obtain the final values.