# A Data Throughput Prediction Using Scheduling And Assignment Technique

**M.Rajarajeswari [1], P.R.Kandasamy[2], T.Ravichandran[3]**
1. Research Scholar, Dept. of Mathematics Karpagam University,Coimbatore.
2.Professor and Head, Dept. of M.CA Hindusthan Institute of Technology,coimbatore.
3.The Principal, Hindusthan Institute of Technology, coimbatore.

**Abstract:**
Task computing is a computation to fill the gap between tasks (what user wants to be done), and services (functionalities that are available to the user). Task computing seeks to redefine how users interact with and use computing environments. Wide distributed many-task computing (MTC) environment aims to bridge the gap between two computing paradigms, high throughput computing (HTC) and high-performance computing (HPC). In a wide distributed many-task computing environment, data sharing between participating clusters may become a major performance constriction. In this project, we present the design and implementation of an application-layer data throughput prediction and optimization service for many-task computing in widely distributed environments using Operation research. This service uses multiple parallel TCP streams which are used to find maximum data distribution stream through assignment model which is to improve the end-to-end throughput of data transfers in the network. A novel mathematical model (optimization model) is developed to determine the number of parallel streams, required to achieve the best network performance. This model can predict the optimal number of parallel streams with as few as three prediction points (i.e. three Switching points). We implement this new service in the Stork Data Scheduler model, where the prediction points can be obtained using Iperf and GridFTP samplings technique. Our results show that the prediction cost plus the optimized transfer time is much less than the non optimized transfer time in most cases. As a result, Stork data model evaluates and transfer jobs with optimization service based sampling rate and no. of task is given as input, so our system can be completed much earlier, compared to non optimized data transfer jobs.

**Key words**: Optimization, Assignment Technique, Stork scheduling Data throughput.

**Modules:**
1) Construction of Grid Computing Architecture.
2) Applying Optimization Service.
3) Integration with Stork Data cheduler.
4) Applying Quantity Control of Sampling Data.
5) Performance Comparison.

## Existing System:

**TCP** is the most widely adopted transport protocol but it has major bottleneck. So we have gone for other different implementation techniques, in both at the transport and application layers, to overcome the inefficient network utilization of the TCP protocol. At the transport layer, different variations of TCP have been implemented to more efficiently utilize high-speed networks. At the application layer, other improvements are proposed on top of the regular TCP, such as **opening multiple parallel streams** or **tuning the buffer size**. **Parallel TCP streams** are able to achieve high network throughput by behaving like a single giant stream, which is the combination of n streams, and getting an unfair share of the available bandwidth.

## Disadvantage Of System:

- In a widely distributed many-task computing ernvionment, data communication between participating clusters may become a major performance bottleneck.
- TCP to fully utilize the available network bandwidth. This becomes a major bottleneck, especially in wide-area high speed networks, where both bandwidth and delay properties are too large, which, in turn, results in a large delay before the bandwidth is fully saturated.
- Inefficient network utilization.

**Proposed System:**

We present the design and implementation of a service that will provide the user with the optimal number of parallel TCP streams as well as a provision of the estimated time and throughput for a specific data transfer. A novel mathematical model (optimization model) is developed to determine the number of parallel streams, required to achieve the best network performance. This model can predict the optimal number of parallel streams with as few as three prediction points (i.e. three Switching points). We implement this new service in the Stork Data Scheduler model, where the prediction points can be obtained using Iperf and GridFTP samplings technique.

**Advantage of System:**

- The prediction models, the quantity control of sampling and the algorithms applied using the mathematical models.
- We have improved an existing prediction model by using three prediction points and adapting a full second order equation or an equation where the order is determined dynamically. We have designed an exponentially increasing sampling strategy to get the data pairs for prediction
- The algorithm to instantiate the throughput function with respect to the number of parallel streams can avoid the ineffectiveness of the prediction models due to some unexpected sampling data pairs.
- We propose to find a solution to satisfy both the time limitation and the accuracy requirements. Our approach doubles the number of parallel streams for every iteration of sampling, and observes the corresponding throughput.
- We implement this new service in the Stork Data Scheduler, where the prediction points can be obtained using Iperf and GridFTP samplings

## Implementation module:

In this project we have implemented the optimization service, based on both Iperf and GridFTP. The structure of our design and presents two scenarios based on both, GridFTP and Iperf versions of the service. For the GridFTP version, these hosts would have GridFTP servers. For the Iperf version, these hosts would have Iperf servers running as well as a small remote module (TranServer) that will make a request to Iperf. Optimization server is the orchestrator host, designated to perform the optimization of TCP parameters and store the resultant data. It also has to be recognized by the sites, since the third-party sampling of throughput data will be performed by it. User/Client represents the host that sends out the request of optimization to the server. All of these hosts are connected via LAN. When a user wants to transfer data between two site , the user will first send a request that consists of source and destination addresses, file size, and an optional buffer size parameter to the optimization server, which process the request and respond to the user with the optimal parallel stream number to do the transfer. The buffer size parameter is an optional parameter, which is given to the GridFTP protocol to set the buffer size to a different value than the system set buffer size. At the same time, the optimization server will estimate the optimal throughput that can be achieved, and the time needed to finish the specified transfer between two site. This information is then returned back to the User/Client making the request. Stork is a batch scheduler, specialized in data placement and movement. In this implementation, Stork is extended to support both estimation and optimization tasks. A task is categorized as an estimation task, if only estimated information regarding to the specific data movement is reported without the actual transfer. On the other hand, a task is categorized as optimization, if the specific data movement is performed, according to the optimized estimation results.

## Mathematical Model :

A novel mathematical model (optimization model) is developed to determine the number of parallel streams, required to achieve the best network performance. This model can predict the optimal number of parallel streams with as few as three prediction points (i.e. three Switching points).We propose to find a solution to satisfy both the time limitation and the accuracy requirements. Our approach doubles the number of parallel streams for every iteration of sampling, and observes the corresponding throughput. While the throughput increases, if the slope of the curve is below a threshold between successive iterations, the sampling stops. Another stopping condition is: if the throughput decreases compared to the previous iteration before reaching that threshold.

**Assignment Problem:**

Consider an matrix with n rows and n columns, rows will be consider as grid let and columns will as jobs.
Like,

$$\begin{array}{cc} & \begin{array}{cccc} \text{Job 1} & \text{Job 2} & \ldots\ldots. & \text{Job n} \end{array} \\ \begin{array}{c} \text{Grid 1} \\ \text{Grid 2} \\ \ldots\ldots. \\ \text{Grid n} \end{array} & \left[\begin{array}{cccc} \text{Task} & \text{Task2} & \ldots & \text{Task} \\ \text{Tas} & \text{Task} & \ldots & \text{Task} \\ & & & \\ \text{Tas} & \text{Task} & \ldots & \text{Task} \end{array}\right] \end{array}$$

There will be more than one job for each grid so assign problem occur. to solve this problem we are going for new mathematical model to solve this problem. Three condition occur

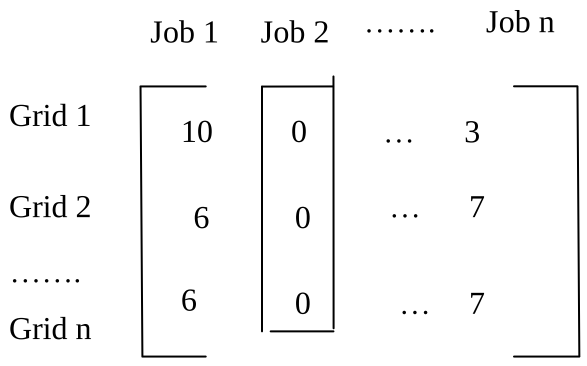
1) Find out the minim value of each row and subtract least value with same row task value. Make least value as zero. If process matrix diagonally comes as “0” then process stop and job will assign successfully and job assign successfully.

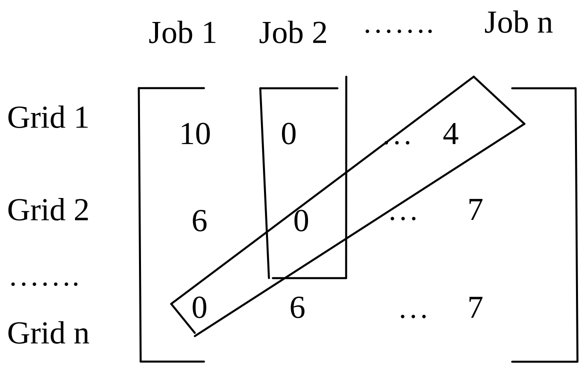
$$\begin{array}{cc} & \begin{array}{cccc} \text{Job 1} & \text{Job2} & & \text{Job n} \end{array} \\ \begin{array}{c} \text{Grid} \\ \text{Grid} \\ \ldots\ldots. \\ \text{Grid} \end{array} & \left[\begin{array}{cccc} 10 & 3 & \ldots & \boxed{0} \\ 6 & \boxed{0} & \ldots & 7 \\ & & & \\ \boxed{0} & 6 & \ldots & 7 \end{array}\right] \end{array}$$

2) Find out the minim value of each row and Subtract least value with same row task value. Make least value as zero. if column wise matrix diagonally comes as “0” then , then Find out minim value of each column and subtract least value with same row column value. Make least value as zero. Then if process matrix diagonally comes as “0” then process stop and job will assign successfully and job assign successfully.

$$\begin{array}{cc} & \begin{array}{cccc} \text{Job 1} & \text{Job 2} & \ldots\ldots. & \text{Job n} \end{array} \\ \begin{array}{c} \text{Grid 1} \\ \text{Grid 2} \\ \ldots\ldots. \\ \text{Grid n} \end{array} & \left[\begin{array}{cccc} 10 & 0 & \ldots & 3 \\ 6 & 0 & \ldots & 7 \\ & & & \\ 6 & 0 & \ldots & 7 \end{array}\right] \end{array}$$

3) Find out the minim value of each row and subtract least value with same row task value. Make least value as zero. If column wise matrix diagonally comes as “0” then, then Find out minim value of each column and subtract least value with same row column value. Make least value as zero. Then if process matrix diagonally will comes as “0” then process stop and that job will be discard .

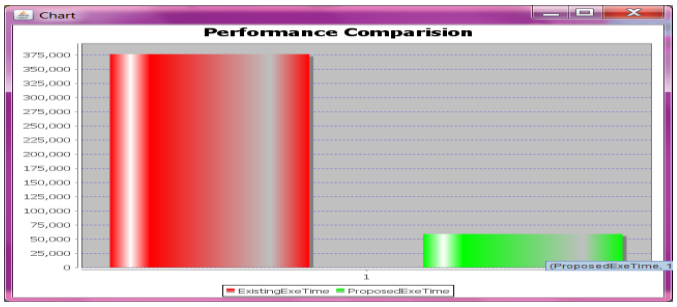


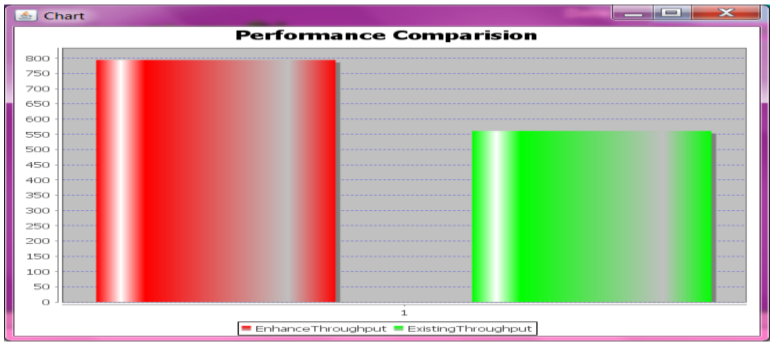


Experimental results:

| Test Scenario | Pre-Condition | Test Case | Expected Output | Actual Output | Result |
|---|---|---|---|---|---|
| Applying Optimization Service | Client send request to EOS server then that request will be transfer to Gridftp server | Check the client request will be Valid request or not(only txt format file can be download/Estimate from Gridftp server ) | Client Requested will be response to client with some information (number of stream taken to transfer the file and throughput achieve at the time of transfer file) | File received successfully | Pass |
| Integration with Stork Data Scheduler | If More than one Client make request to EOS server | Stork data scheduler make schedule for the incoming request exceed than one request/ checking the is Estimation or optimization Of service | Priority will be assigned to each user request(first come first process) / if estimation request will be give most first priority | According to priority the request process in Gridftp server and response given to client | Pass |

**Performance Comparison**





## Conclusion:

This study describes the design and implementation of a network throughput prediction and optimization service for many-task computing in widely distributed environments. This involves the selection of prediction models, the mathematical models. We have improved an existing prediction model by using three prediction points and adapting a full second order equation, or an equation where the order is determined dynamically. We have designed an exponentially increasing sampling strategy to get the data pairs prediction. We implement this new service in the Stork Data Scheduler, where the prediction points can be obtained using Iperf and GridFTP samplings. The experimental results justify our improved models as well as the algorithms applied to the implementation. When used within the Stork Data Scheduler, the optimization service decreases the total transfer time for a large number of data transfer jobs submitted to the scheduler significantly compared to the non optimized Stork transfers.

## Reference:

[1] I. Raicu, I. Foster, and Y. Zhao, "Many-Task Computing for Grids and Supercomputers," Proc. IEEE Workshop Many-Task Computing on Grids and Supercomputers (MTAGS), 2008.
[2] Louisiana Optical Network Initiative (LONI), http://www.loni.org/, 2010.
[3] Energy Sciences Network (ESNet), http://www.es.net/, 2010. [4] TeraGrid, http://www.teragrid.org/, 2010.
[5] S. Floyd, "Rfc 3649: Highspeed TCP for Large Congestion Windows," 2003.
[6] R. Kelly, "Scalable TCP: Improving Performance in Highspeed Wide Area Networks," Computer Comm. Rev., vol. 32, Experiments," IEEE Network, vol. 19, no. 1, pp. 4-11, Feb. 2005. no. 2, pp. 83- 91, 2003.
[7] C. Jin, D.X. Wei, S.H. Low, G. Buhrmaster, J. Bunn, D.H. Choe, R.L.A. Cottrell, J.C. Doyle, W. Feng, O. Martin, H. Newman, F. Paganini, S. Ravot, and S. Singh, "Fast TCP: From Theory to Networks," Computer Comm. Rev., vol. 32, Experiments," IEEE Network, vol. 19, no. 1, pp. 4-11, Feb. 2005.
[8] H. Sivakumar, S. Bailey, and R.L. Grossman, "Psockets: The Case for Application-Level Network Striping for Data Intensive Applications Using High Speed Wide Area Networks," Proc.
IEEE Super Computing Conf. (SC '00), p. 63, Nov. 2000.
[9] J. Lee, D. Gunter, B. Tierney, B. Allcock, J. Bester, J. Bresnahan, and S. Tuecke, "Applied Techniques for High Bandwidth Data Transfers across Wide Area Networks," Proc. Int'l Conf. Computing in High Energy and Nuclear Physics (CHEP '01), Sept. 2001.
[10] H. Balakrishman, V.N. Padmanabhan, S. Seshan, and R.H.K.M. Stemm, "TCP Behavior of a Busy Internet Server: Analysis and Improvements," Proc. IEEE INFOCOM '98, pp. 252-262, Mar. 1998.
[11] T.J. Hacker, B.D. Noble, and B.D. Atley, "Adaptive Data Block Scheduling for Parallel Streams," Proc. IEEE Int'l Symp. High Performance Distributed Computing (HPDC '05), pp. 265-275, July 2005.
[12] L. Eggert, J. Heideman, and J. Touch, "Effects of Ensemble TCP," ACM Computer Comm. Rev., vol. 30, no. 1, pp. 15-29, 2000.
[13] R.P. Karrer, J. Park, and J. Kim, "TCP-ROME: Performance and Fairness in Parallel Downloads for Web and Real Time Multimedia Streaming Applications," technical report, Deutsche Telekom Labs, 2006.
[14] D. Lu, Y. Qiao, and P.A. Dinda, "Characterizing and Predicting TCP Throughput on the Wide Area Network," Proc. IEEE Int'l Conf. Distributed Computing Systems (ICDCS '05), pp. 414-424, 2005.
[15] E. Yildirim, D. Yin, and T. Kosar, "Prediction of Optimal Parallelism Level in Wide Area Data Transfers," to be published in IEEE Trans. Parallel and Distributed Systems, 2010.