M.I.T. LABORATORY FOR COMPUTER SCIENCE February 21, 1978

Computer Systems Research Division Request for Comments No. 168

DISTRIBUTED SYSTEMS: MODELING AND PERFORMANCE EVALUATION

by Liba Svobodova

In the recent past I have been asked several times about the research possibilities in the area of modeling and performance evaluation of distributed systems, and in particular, of distributed data bases. Thus, I thought it might be worthwhile to have an answer to this question summarized in the form of a working paper for the benefit of other interested parties.

The work on modeling and performance evaluation in distributed systems can be divided into several categories. I will exclude some of these categories right from the beginning. One such category is performance evaluation of low level protocols, for example, evaluation of the ethernet or the ringnet, or even protocols such as TCP and the question of routing and capacity selection in store-and-forward networks. The second area is so called load sharing where jobs that arrive at a specific computer system may be forwarded to another node because the receiving node is temporarily overloaded. In fact, I will concentrate primarily on the problem of distributed data bases.

From the performance evaluation point of view, probably the oldest and the most explored area of distributed systems is the problem of file allocation [AKOK77, CASE72, CHU69, LEVI75]. The models used in this area can be classified as optimization models where the problem is to minimize cost or response time, given the cost of storage at individual nodes, the cost of communication and the pattern of accessing individual files. The newer models takes into consideration duplication of individual files. However, these models in general are not concerned with the problem of update where more than one copy of a file exists. As we know from the research dealing with the problem of a multiple-copy update in distributed systems, the update algorithms can be rather complex and may cause a significant overhead. The models for optimal allocation of files should take this into consideration. However, even without this consideration, such optimization models are quite complicated and often unsolvable for a general situation. An interesting contribution in this area is a set of memos by Allen Luniewski [LUNI77, LUNI78] that explored the advantages of a broadcast network. A broadcast network not just makes the solution simpler, but the solution can change dynamically, that is, the network can decide dynamically as to which files should be replicated and where they should be located. However, the multiple-copy update is modeled only in a very simple manner.

As already alluded, the problem of multiple copies in a distributed system needs thorough performance analysis. Some of the interesting algorithms turn out to be quite impractical when the performance is considered, especially when a realistic view of the real world is taken where individual nodes or individual links may fail. Very little work has been done in this area. Individual authors of different algorithms for consistent update of multiple copies in a distributed system specify performance of their algorithms in terms of the number of messages that have to be exchanged. However, the number of messages may be a very misleading measure if the amount of processing that has to be done in individual nodes and the amount of waiting for the information needed to make a decision is significant. Probably the most comprehensive analysis of this problem has been done by Hector Garcia-Molina [GARC78A]. In this particular piece of research, Garcia simulated a centralized update algorithm and compared it with Thomas' algorithm which is completely distributed. It turned out that the centralized algorithm was always better than the distributed algorithm in the situation without failures. The centralized algorithm can be made robust by using a modified two-phase commit protocol that uses a majority vote in deciding whether an operation should be committed [GARC78B]. Since failures in a distributed system should be rare, it is important that the chosen algorithm performs as efficiently as possible in a nonfailure situation; when a failure does occur, the algorithm has to be robust enough to recovery from it, but the time may not be as critical as for normal operation. However, the degree of robustness afforded by different schemes is even harder to assess than their normal performance.

Probably the most important area is the one that deals with the problem of partitioning of computations and data bases in a distributed system. Algorithms have been proposed for dealing with a distributed update, that is, an update where different pieces of information that have to be changed as part of an atomic operation reside at different nodes in a computer network. Of course, if the number of nodes that have to be involved in an atomic update is large, the cost of such an update will be significantly higher than if the udpate can concentrate on one or just a small collection of nodes. On the other hand, distribution of data and computing power can greatly reduce response time for strictly local operations. Also, paritioning is an important means of confining errors, and achieving the goal of partial operability, where the system is capable of carrying on at least a subset of the normally provided services in spite of various failures within the system [MONT78]. Thus, this is one example where proper partitioning will play an important role, but where the tradeoffs are far from obvious. Another example of a partitioning problem is the problem of the so called distributed query. It may be necessary to compare, record by record, two or more files that reside in different nodes. In such a situation it may be important in what order the query is processed, because the amount of information that will have to be transferred between nodes may vary significantly. Some work in this area has been done at CCA in connection with SDD-1 [REEV78, WONG77], by the INGRESS people (UC Berkeley) and by the System R people (IBM San Jose Research Laboratory).

The basic problem with distributed processing is that it may be impossible to have, at any time, a complete information about the global state of the system. Thus, individual nodes may need to process whatever requests they get with only an estimate of what the rest of the system is doing. A manifestation of this problem can be found on many different levels, starting with the "two generals" problem in the lowest communication protocols. In addition to this problem of reaching a mutual agreement when the only possible communication is over unreliable channels, or, better, even if failures are not a problem, the cost of maintaining complete and always the most current information about the global state of some application may be prohibitive. In many cases, it may be possible (and perhaps necessary) to take advantage of the semantics of the particular application, that is, to design algorithms, at the application level, that never require a consistent snapshot of the global state of the application. Victor Lesser at the University of Massachusetts has been working on this problem in the context of AI-like problem solving systems [LESS77]. But these new algorithms designed to deal with the problem of uncertainty about the global state of the system need to be evaluated and also compared with completely centralized algorithms for situations where centralized processing is a viable solution.

One of the most interesting and also the most difficult aspects of modeling and performance evaluation efforts is the evaluation of efficiency vs reliability trade-offs. A gross example of such a trade-off is how many copies of individual files should be supported, and more subtle problems are what algorithm should be used for updating these copies. I discussed the difference between a centralized, that is, master-backup copy arrangement,and a completely distributed algorithm where any copy can be updated in the DSG progress report [DSG78]. However, the problem is more subtle. It spans the communication protocols starting from the very low ones up to the time-out conventions that can be specified by the designer of an application. Although some modeling can be helpful in such a total system evaluation effort, experimentation with a prototype system is undoubtedly the required path.

To summarize, understanding the performance possibilities and tradeoffs of distributed systems is very important, but I don't believe that queuing theory will be very helpful. Because of the complexity of the tradeoffs involved, the first and a very important step is to get a clearer understanding of the requirements. Once the requirements are specified, the optimal solution probably is not the one that gives the best performance, but the simplest solution that satisfies the requirements.

REFERENCES:

AKOK77 Akoka, J., Chen, P., "Optimization of Distributed Database Systems and Computer Networks," M.I.T. Sloan School of Management, WP916-77, March 1977.

CASE72 Casey, R.G., "Allocation of Copies of a File in an Information Network," Proc. of AFIPS SJCC 1972, pp. 617-625.

CHU69 Chu, W.W., "Optimal File Allocation in a Multicomputer Information Center," IEEE Transactions on Computers, Vol. C-18, No. 10, October 1969, pp. 885-889.

DSG78 Clark, D.D., et al., "Semantics of Distributed Computing: Progress Report of the Distributed Systems Group," M.I.T., Laboratory for Computer Science, September 20, 1978.

GARC78A Garcia-Molina, H., "Performance Comparison of Update Algorithms for Distributed Databases," Stanford University, Digital Systems Laboratory, Technical Note No. 143, June 1978.

GARC78B Garcia_Molina, H., "Performance Comparison of Update Algorithms for Distributed Databases: Crash Recovery in the Centralized Locking Algorithm," Stanford University, Digital Systems Laboratory, Progress Report No. 7, November 1978.

LESS77 Lesser, V.R., "The Application of Artificial Intelligence Techniques To Cooperative Distributed Processing," National Science Foundation Proposal, 1977.

LEVI75 Levin, K.D., Morgan, H.L., "Optimizing Distributed Databases: A Framework for Research," Proc. of AFIPS NCC, 1975, pp. 473-478.

LUNI77 Luniewski, A.W., "File Allocation in a Distributed System," M.I.T. Laboratory for Computer Science, Computer Systems Research Division, Request for Comments No. 152, December 19, 1977.

LUNI78 Luniewski, A.W., "Some Results on File Allocation in a Local Network," M.I.T. Laboratory for Computer Science, Computer Systems Research Division, Request for Comments No. 162, March 22, 1978.

MONT78 Montgomery, W.A., "Robust Concurrency Control for a Distributed Information System," M.I.T. Department of Electrical Engineering and Computer Science, PhD Thesis, November 1978.

REEV78 Reeve, C.L., Wong, E., Rothnie J.B., "Query Optimization Algorithm for SDD-1: A System for Distributed Databases," Computer Corporation of America, January 1978.

WONG77 Wong, E., "Retrieving Dispersed Data from SDD-1: A System for Distributed Database," Proc. of Second Berkeley Workshop on Distributed Data Management and Computer Networks, Berkeley, California, May 1977, pp. ??