Optimizing Configuration for Hierarchical Storage Based Continuous Media Server

Youjip Won
Server Architecture Lab
Intel Corp.
yjwon@co.intel.com

Jaideep Srivastava and Zhi-Li Zhang
Computer Science Dept
University of Minnesota
{srivasta|zhzhang}@cs.umn.edu

1 Introduction

Recent advances in computing and communication technologies have made it technically feasible and economically viable to provide on-line access to a variety of information services over high speed networks. Particularly, convergence of various technological factors, namely in network access and in video coding and transmission, have recently brought a rapid growth of interest in on-line access to *multimedia* services. The problem of large-scale provision of services has several implications for the design of a storage server. Voluminous nature of a multimedia file is one of the primary reasons which makes the design of the server non trivial. With MPEG-2 compression technique a movie file of 110 min length requires more than 3 G bytes of storage space. Storing thousands of files of this size on a disk subsystem requires huge amounts of disk space. According to user surveys[7], file-access frequency is strictly biased in favor of a small number of popular titles, while the rest of the files are rarely accessed in commercial video rental. This characteristic of the user-access pattern enables the storage architecture designer to exploit the hierarchical storage structure in managing a large volume of information.

The advantage of using hierarchical storage architecture is to exploit the popularity or *hotness* of a file and assign space the appropriate storage hierarchy to each file, thereby maximizing the cost-performance ratio. The efficiency of hierarchical storage architecture is maximized when the capacity of each hierarchy is well balanced and the behavior of each storage hierarchy is well harmonized. Thus, in configuring hierarchical storage, these system parameters have to be carefully taken into account.

From the server's point of view, maximizing throughput is primary concern in various aspects of the design. Throughput can be thought as the number of requests which can be handled or *serviced* by the server per unit time. When user request needs to be serviced immediately, the number of requests which can be handled per unit time, *throughput* is inversely proportional to the probability that the incoming request is blocked. Thus, *Blocking probability* is an important concern in achieving a desired cost-effectiveness. In hierarchical storage architecture which consists of tertiary storage and secondary storage, the request blocking can be due to congestion in tertiary storage or due to congestion in secondary storage

In this work, the effort is concentrated on finding a minimum amount of storage resources for each hierarchy to achieve a given throughput. Storage capacity and bandwidth capacity of a unit storage device, e.g. *a disk drive*, *a tape drive* is fixed. *Blocking probability* is a metric for throughput.

There have been a number of studies about using hierarchical storage in continous media server[2, 4, 6, 5, 9]. Also, extending the notion of *hierarchy* to distributed system has been propsed[1, 10]. However, none of these literatures have dealt with the issues of *What capacity for each storage hierarchy is required to satisfy given performence metric*.

2 Problem Formulation

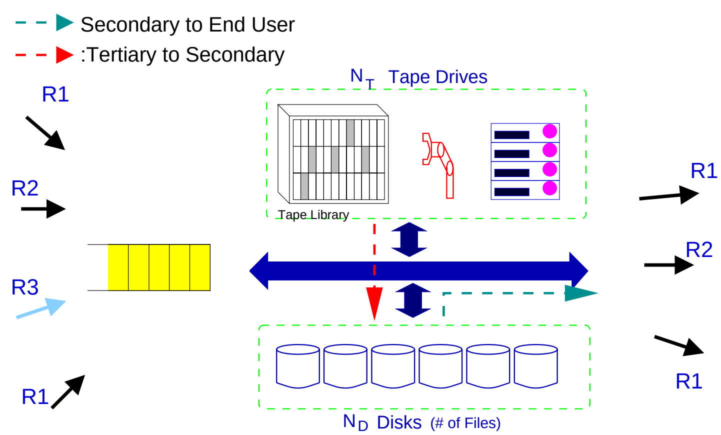

Figure 1: Service Mechanism

Fig. 1 illustrates the hierarchical storage architecture and its service mechanism. When requests arrive at the server, some of them cannot be serviced immediately due to limitation on system resources. All files reside permanently in the tertiary storage, which can be in the form of tape cartridges. A file is loaded from the tape library to the disk in an *on-demand* basis. When a user request arrives at the server, a mechanical device loads the respective tape cartridge into the tape drive unless it is already loaded on the disk or being loaded by the preceeding request. Then, the requested file is loaded from the tertiary storage into the secondary storage, which is a disk subsystem and then is sent to the user. It is also possible that incoming request observes that the requested file is either in the tape drive being loaded on the disk or loaded on the disk. In former situation, the request waits until the file is loaded on the disk and then starts to be serviced to user. In latter case, user is serviced immediately. In case all the tape drives are transferring files to the disk or disk subsystem is full, the server cannot accommodate new file, *Blocking situation*. Detailed queuing analysis of hierarchical storage behavior can be found in Won[8].

The objective of this work is to find minimum cost storage capacity to satisfy the given performance metric. The performance metric adopted in this work is *Blocking Probability*, probability that the incoming request is not serviced immeidiately. Given the performance specification of the individual disk subsystem and tape drive, e.g. *read/write bandwidth*, *head movement overhead*, we determine the amount of storage resources to preserves the blocking probability.

3 Approach

We develop an efficient methodology to find the optimal solution for configuration problem. In our hierarchical storage structure, server consists of two storage hierarchies - secondary and tertiary storage. Request is blocked when either of the hierarchies cannot accommodate the incoming request. Hierarchical storage system is modeled using the closed queuing network to compute the blocking probability. The customer in the closed queuing system corresponds to the files in the tape library. Thus, each customer repeats the cyclic transition among *Idle*, *On Tape Drive*, and *On Disk* status. The figure in the bottom of Fig. 2 illustrates the closed queuing network representation of the hierarchical storage system.

Given the blocking probability, e.g. δ in Fig. 2, it is of interest to determine the lower bound of the number of tape drives or the capacity of the disk subsystem to achieve the blocking probability. Fig. 2 illustrates the approach to find the configuration of storage

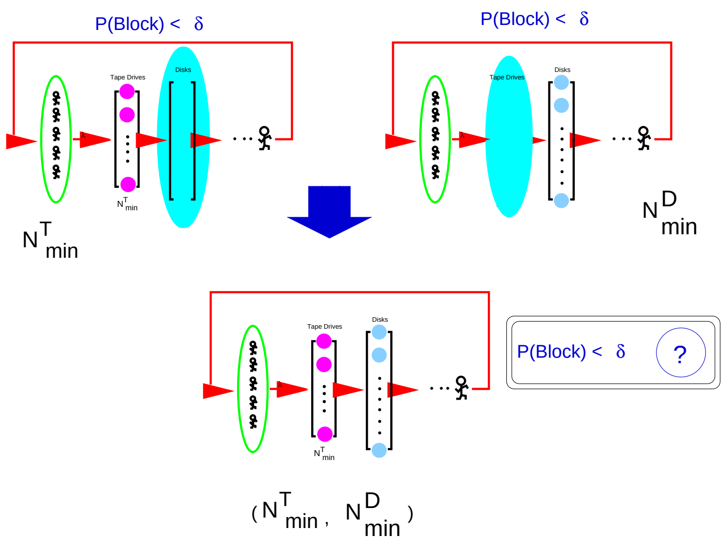

Figure 2: Partial Optimal Configuration and its Integration

hierarchy via partial optimal configuration and its integration. The configuration methodology is described in the following.

Given the user access pattern, aggregate user request arrival rate, data rate of the tape drive and data rate of the disk,

Step 1 Find minimum number of tape drives given that there is sufficient capacity secondary storage with blocking probability $\mathcal{P}$.

Step 2 Find minimum capacity of the disk subsystem given that there is sufficient tape drives with blocking probability $\mathcal{P}$.

Step 3 Build a hierarchical storage system with the amount of resources found in Step 1 and Step 2.

The amount of resources found in Step 1 and Step 2 is called *Partial Optimal Configuration*. Obtaining the partial optimal configuration is an import problem to make the entire framework useful. Cohen[3] found the famous result that in *Quasi Random Input, loss* *system*, the probability for the server state is determined by the *idle time* arrival rate and *expected service time*. Based on the Cohen's framework, we developed queuing system which precisely models the system where one storage hierarchy has sufficient capacity[8].

We use partial optimal configuration for disks and tape drives to obtain the optimal configuration. However, the server with N_T^{min} number of tape drives and N_D^{min} of disks may not preserve the blocking probability δ. This is because N_T^{min} and N_D^{min} is obtained given that there are sufficient amount of disks and sufficient amount of tape drives, respectively. Algorithm Opt_Config in table 1 is used to find the optimal storage hierarchy for a given probability δ.

Algorithm 1 Opt_Config

▷ *Finding Optimal Configuration for given* $\mathcal{P}$
$Opt\_Config(C_T, C_D, \Im, \mu_T, \mu_D, \mathcal{P})\{$
    Find partial optimal configuration $\Re_D$ with respect to blocking probability $\mathcal{P}$;
    Find partial optimal configuration $\Re_T$ with respect to blocking probability $\mathcal{P}$;
    Let starting configuration $\Re$ be $\langle N_T^{min}, N_D^{min}, \mu_T, \mu_D \rangle$;
    $N_T \leftarrow N_T^{min}$;
    $N_D \leftarrow N_D^{min}$;
    $C \leftarrow C_T N_T + C_D N_D$;
    while $(\mathcal{F}(N_T, N_D, \mu_T, \mu_D) > \mathcal{P})\{$
        ▷ Find the next closest point to $C_T N_T + C_D N_D = C$
        for $X \leftarrow N_D^{min}$ to $\lfloor \frac{C-(N_T-N_T^{min})C_T}{C_D} \rfloor + N_D$;
            $Y \leftarrow \lfloor \frac{C-C_D(X-N_D)}{C_T} \rfloor + N_T + 1$;
            ▷ Compute Distance between (X, Y) and $N_T C_T + N_D C_D = C$
            $Dist \leftarrow \frac{|C_D X + C_T Y - C|}{\sqrt{C_T^2 + C_D^2}}$;
            if $(Dist < MinDist)$ $\{$
                $MinDist \leftarrow Dist$;
                $X^{min} \leftarrow X$;
                $Y^{min} \leftarrow Y$;
            $\}$
        $\}$
        $N_D \leftarrow X^{min}$;
        $N_T \leftarrow Y^{min}$;
        $MinDist \leftarrow \infty$;
    $\}$
▷ (N_T, N_D) is optimal configuration
return (N_T, N_D) ;
$\}$

Table 1: Algorithm for Opt_Config

4 Summary

Hierarchical storage system is being proposed to effectively utilize the popularity of each file providing cost effective solution for massive data storage system. Multimedia application is one of the context where data volume is much larger than text based application. In this work, we are interested in finding optimal configuration of storage hierarchy while maintaining a certain level of throughput, *blocking probability*. Given the speed of the tape drives and disk data rate, capacity of the hierarchical storage architecture is represented by the number of tape drives, N_T and the capacity of the secondary storage, N_D. Optimal configuration is a minimum number of N_T and N_D with a given data rate at each storage hierarchy and user access characteristics. In designing a hierarchical storage architecture, the user access profile needs to be carefully incorporated to avoid waste of the resources. In the method developed in this work, we first find the partial optimal configuration for each storage hierarchy. Then, from the integration of the partial optimal configuration, we obtain the optimal configuration. Queuing representation of the original hierarchical storage architecture is a two stage queuing network with blocking. By assuming the sufficient capacity in one storage hierarchy, the queuing model of the hierarchical storage system becomes single stage queuing model which is *finite capacity*, *finite customer*, *proper loss* queuing system. Cohen's result for *Generalized Engset Formula*[3] enables us to build a fine queuing model for partial configuration. The resulting analytical model for partial configuration becomes much simpler than original two stage queuing network to analyze. In this work, we propose an algorithm and its probabilistic framework to find a optimal configuration by integrating the partial optimal configuration.

References

[1] D. W. Brubeck and L. A. Rowe. Hierarchical storage management in a distributed VOD system. *IEEE Multimedia Magazine*, 3(3):37–47, Fall 1996.

[2] A. L. Chervenak. *Tertiary Storage: An Evaluation of New Applications*. PhD thesis, University of California at Berkeley, December 1994.

[3] H. W. Cohen. The Generalized Engset Formulae. *Philips Telecommunication Review*, 18(4):158–170, November 1957.

[4] Y. N. Doganata and A. N. Tantawi. A cost/performance study of video servers with hierarchical storage. In *Int'l Conf. on Multimedia Computing and Systems*, Piscataway, N.J., 1994. IEEE Press.

[5] Ghandeharizadeh, S. and Shahabi, Cyrus. On Multimedia Repositories, Personal Computers, and Hierarchical Storage. In *Conf. Proc. ACM Multimedia*, pages 407–416, 1994.

[6] W. T. M. Kienzle, D. Sitaram. Using a storage hierarchy in movie-on-demand servers. Technical report, IBM T.J. Watson Research Center, Hawthorne, NY, 1994.

[7] V. S. Magazine. Distributions of video viewing frequencies. Video Store Magazine, Dec. 13 1992.

[8] Y. Won. *Issues in Designing Distributed Hierarchical Storage System For Continuous Media Service*. PhD thesis, Department of Computer Science, University of Minnesota, August 1997.

[9] Y. Won and J. Srivastava. Parametric Evaluation of Performance Behavior in Hierarchical Storage Architecture. *Submitted to International Journal of Performance Evaluation*, 1997.

[10] Y. Won and J. Srivastava. Strategic Replication of Video Files in a Distributed Environment. *International Journal of Multimedia Tools and Application*, To Appear.