Chapter 5

Modeling and Optimization of a Multiresolution Image Retrieval System [1]

Contents

5.1 Introduction

This chapter deals with a different scenario for adaptivity. In Chapter 1 we had defined adaptivity as the possibility of choosing different codewords for a given block *depending on the context*. While in Chapters 2-4 we considered video sequences, where the component frames are to be transmitted in the order they were generated, here we consider a *set of images* that is being accessed remotely. The images are now accessed in a random order and a multiresolution scheme is used to transmit them. We will further depart from the approach taken in previous chapters in that

[1]This chapter represents joint work with Zhensheng Zhang. For related publications see [78, 77]

(a) we will be optimizing the performance based on a model and (b) the aim of the optimization will be a delay rather than the R-D performance. The adaptivity here comes from the fact that the set of images will be considered globally to determine the allocation for individual images.

Consider a generic multiresolution (MR) remote image retrieval system (see [26] for an example of such a system). The multiresolution approach is already being used for commercial products (e.g. Kodak's Photo CD) and has also been proposed for retrieval of video [18]. Users accessing the system will be searching for one or more images within those available in the remote database. The two main components of the system are an image database and a user interface which handles the communication resources transparently to the user. We assume that there are two main stages in a query: (i) the *database search* stage, where in response to the user specification the database manager defines a set of possible candidate images, and (ii) the *browsing* stage, where the user tries to select one or more candidate images, called *target images*. In the latter stage the user is presented with a set of low resolution images (e.g. icons), and can then view them at increasing resolutions, up to the highest available quality, and this until one or more images are selected or the query is terminated. The motivation is that by having fast access first to "coarse" versions of the images, users are allowed to discard, if desired, some of the images *without necessarily having to receive the full quality image*, thus reducing the overall transmission costs of the system. When favoring an MR approach, the underlying assumption is that *the communication costs are the limiting factor*. This situation arises either because (i) the users have access to low–speed (or shared) links, so that transmission delay dominates the total delay in the query (over, for instance, the delay introduced by the search within the database), or simply because (ii) the system has to be designed to minimize the total transmission cost, which we assume to be proportional to the transmission time.

In this chapter we will concentrate on the browsing stage of the queries. We will further assume that browsing and database search are independent so that our optimization of the browsing stage will not affect the performance of the database search stage. While work reported in the literature has focused on the progressive image transmission schemes [53, 96, 64] here we look at the image coding scheme from a systems perspective. Images in the database are coded with an MR scheme (which we do not specify) so that, taking the two-resolution case as an example, at the start of the browsing stage a fraction αB, $0 < \alpha < 1$, of the B bits of the image is transmitted and a low resolution image is reconstructed using those bits. The remaining $(1 - \alpha)B$ needed to reconstruct the full resolution image will only be sent if the user requests it. We tackle the problem of assigning a number of bits to each of the image layers (i.e. in our example choosing α) so that the performance of the image retrieval system is optimized.

Note that in a typical bit allocation problem for an MR image coder [85, 86] the objective is to assign bits to each of the image layers to maximize the full quality and possibly to meet some intermediate quality objectives. However here our concern is to study how the bit allocation among the successive image layers affects the overall system performance. As an example, in [26] arbitrary compression rates are chosen for the different resolutions: we point out that this choice can be made so that the system performance is optimized.

To clarify the scope of our optimization, let us note that we can divide the resources used in an MR image retrieval system into roughly three groups: (i) the database computation resources, (ii) the communication resources, and (iii) the computation (including memory) resources at the user sites. We will only consider the latter two resources, under the assumption that the bit allocation only affects the browsing stage and not the search within the database. We can thus state the problem we are seeking to solve as follows, imposing the constraint that all the images in the database use the same allocation:

Problem 5.1 *How do we allocate the bits to each of the image layers to minimize the total transmission delay or, equivalently, the transmission cost, during a query for a target image.*

At the beginning of the browsing stage, the user is provided with a set of icons from which to select the target image. Since the icons will have very low resolution, it will typically be hard to determine whether the icon set contains a target image and thus the user will have to retrieve some of the images at increasing resolutions in order to make a choice. The trade-off that arises in choosing the bit allocation is clear. If the intermediate resolution were of very high quality (thus requiring a large number of bits, or α close to 1), the user would be able to make a decision on whether the image is acceptable but the cost of retrieving non-acceptable images would be high. Conversely, if the intermediate quality were low (and thus the required bit rate were small, or α close to 0) a decision on the image would not be easy while the cost for choosing a "wrong" icon would be small. The aim of this chapter is to analyze the trade-off.

This chapter is organized as follows. Section 5.2 provides a more detailed description of a multiresolution image retrieval system and formally defines the parameters of the system as well as our objective function. Section 5.3 provides solutions to the problem under the different sets of parameters. In particular, it is shown how a "dynamic" queueing–based approach and a "static" average analysis yield the same results. Section 5.4 draws conclusions and points out areas for further work.

5.2 System definition

5.2.1 Multiresolution browsing

Consider the following flow diagram for the user interaction (see Fig. 5-1). Each user first generates a request and, after a database search, a set of low resolution candidate icons is displayed at the terminal. The user then, at *Stage 1*, selects one of the icons so that its corresponding low resolution image is displayed on the terminal. At *Stage 2*, if (a) the quality of the low resolution image is too poor to decide or (b) the image seems to be adequate for the user requirement, the user requests that the additional information (necessary to create the full resolution picture) is sent (go to *Stage 3*). Otherwise, if the displayed image has sufficient quality and is not one of the targets, it is rejected and another icon is selected (go back to *Stage 1*). At *Stage 3* the full resolution image is displayed and the user can accept it (and terminate the query) or reject it and select another icon (go back to *Stage 1*). The process repeats until an appropriate image is found.

Note that the above description presents a somewhat simplified user interaction since only one candidate image can be considered at any given time. A more general case would not have such a restriction and users would be allowed to store images at different resolutions and then make their decision by comparing those selected. The search can be seen as a process where the user accumulates images at different resolutions (from icon up to full resolution) until the target (one or several images) has been found. While a system with memory might seem more realistic, our results indicate that, as far as the allocation is concerned, the results are identical in both the memory and memoryless cases.

Figure 5-1: Multiresolution image retrieval system: typical user interaction and corresponding system parameters.

5.2.2 System Model

The previous system description can be formalized as follows (refer to Fig. 5-2). Let t be the probability that an image chosen from the set of icons is one of the *target* images. Let α denote the percentage of the image data volume in the low resolution; we assume that all images are coded using the same parameter α. Let $P(\alpha)$ denote the probability that the quality of the image reconstructed using α percent of the bits is sufficient to make a correct decision (see Section 5.2.3). Our objective is to obtain α_{opt}, the optimal value of α such that the mean response time is minimized, where the response time is defined as the time interval from the time the request is generated until the time the target image is found.

We model the user interaction (refer again to Fig. 5-2) by assigning probabilities to the transitions between the successive stages of the query as follows. A transition from *Stage 2* to *Stage 1* occurs when the image has sufficient quality but is not a

Figure 5-2: System model for a multiresolution image retrieval system. t is the probability an image is one of the targets. $P(\alpha)$ is the probability that α percent of the total bits provide sufficient quality. B is the image size.

target, with probability

$$1 - p = P_{2\to 1} = (1-t) \cdot P(\alpha).$$

A transition from *Stage 2* to *Stage 3* occurs if (a) the image has insufficient quality or (b) if a target image has been found, with probability

$$p = P_{2\to 3} = 1 - P(\alpha) + t \cdot P(\alpha).$$

Finally at *Stage 3*, the query will end if a target image has been found and will go back to *Stage 1* otherwise, so that we have:

$$1 - q = P_{3\to 1} = \frac{(1-t)\cdot(1-P(\alpha))}{t \cdot P(\alpha) + 1 - P(\alpha)}, \qquad \text{and}$$

$$q = P_{3\to e} = \frac{t}{t \cdot P(\alpha) + 1 - P(\alpha)}.$$

5.2.3 Probability of sufficient quality

Given a set of N images, $\mathcal{S}$, assume that we allocate to all of them the same α. We propose to model $P(\alpha)$, the probability that an image, picked at random from the set, has "sufficient" quality for the user to make a decision, as follows. To each image from the set $s_j \in S$, we can associate a rate-distortion (R-D) characteristic, where each R-D point corresponds to the image coded at one of the available resolutions. Denote these functions as (R_j, D_j) and suppose they are obtained either through measurements on the image set or based upon a model. Note that here we consider any measure of distortion, in particular, measures based on subjective thresholding are possible. We now define a threshold $D_t(s_j)$, which represents the level of "indistinguishable quality", i.e. increasing the rate to reduce the distortion below that threshold produces virtually no improvement to the subjective quality of the image. Then we can define the normalized rate and distortion functions, δ_j and α_j respectively, as:

$$\delta_j = \min(1, D_t(s_j)/D_j), \quad \text{and} \quad \alpha_j = R_j/R_{max}(s_j), \tag{5.1}$$

where $D_t(s_j)$ can be a common threshold for all images or can be chosen individually for each s_j, and $R_{max}(s_j)$ is the bit rate required by the highest resolution version available for image s_j. Also we define the normalized quality to be $\delta_j = 0$ when no bits are used, i.e. $\alpha_j = 0$.

We can see that $\delta_j(\alpha_j)$ indicates of the likelihood that a given image has sufficient quality. For $\delta(\alpha)$ close to one we are close to the full resolution quality so that we will have sufficient quality for almost any application. For $\delta(\alpha)$ close to zero only "easy" searches (e.g. locating a big object within an image) will be possible, others (e.g. locating a texture) will require increased quality. Once we consider $\mathcal{S}$ as a whole, and given that images are picked at random from the set, we can estimate $P(\alpha)$ as follows:

$$P(\alpha) = \frac{1}{N} \sum_{s_j \in \mathcal{S}} \delta_j(\alpha). \tag{5.2}$$

In the rest of this chapter we will assume that the probability function $P(\alpha)$ is in the form of $1-(1-\alpha)^m$, where m is a positive integer (see Fig. 5-3). Note that our choice is reasonable when considering typical rate-distortion characteristics and it only affects the exact value of our result; the general analysis holds for more general expressions of $P(\alpha)$.

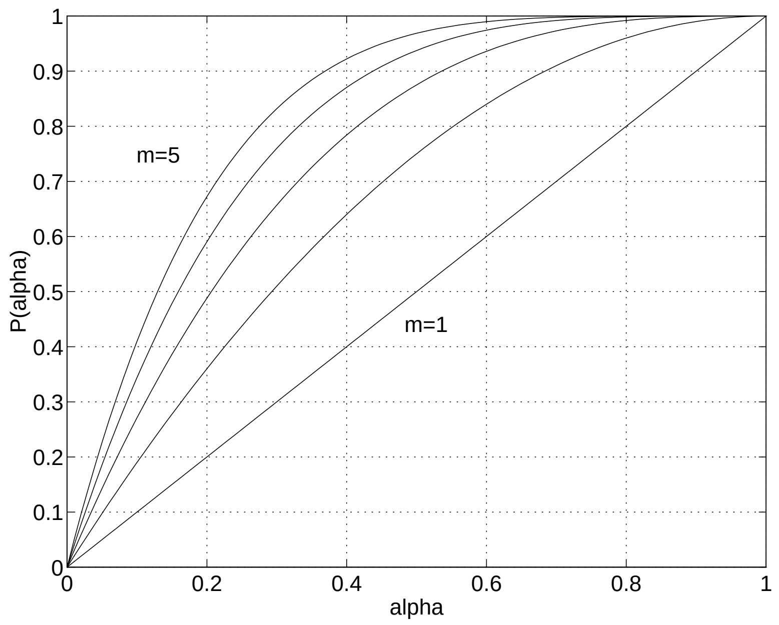

Figure 5-3: Example of $P(\alpha)$ functions for several values of m. Note that the larger m values are more realistic.

5.3 Analysis and Results

We now provide solutions to the optimization problem outlined in the previous section. Note that we formulated the problem of a single user having access to the database but we now consider the possibility of several users sharing the system. Different methods are called for depending on the exact formulation, in particular whether the communication resources are shared or not. However, we will show that as far as the optimal point is concerned, most cases of interest yield the same solution. Note also that here we assume $P(\alpha)$ to be the same for all users accessing the database. This does not mean that all users are supposed to access the same set of images, rather it implies that all image sets are similar as far as their quality-rate trade-off. The problem will be tackled first in a queueing framework (Section 5.3.1) while a static analysis will be proposed later (Section 5.3.2).

5.3.1 Dynamic Analysis

5.3.1.1 Separate channels and large image set

We assume here that the M users in the system do not share the communications resources and have only to share the computation resources of the database. Users generate new requests at rate λ but only when they are idle, i.e. when all previous queries have been completed. The average size of an image is assumed to be $1/\mu_2$. We assume that on average a delay of $1/\mu_1$ is incurred every time a user goes back to Stage 1 in the query; the cost reflects the computation needed in the database to have the next selected image ready for tramission (e.g. encoding, loading into appropriate buffers, etc). Note that although transmission of the icons themselves would produce some delay we are assuming that the icon size is constant and thus we are not including this factor in the optimization process. We assume that the image sets that are being searched are large enough that t is constant during the search. For a given value of α, we are interested in the average response time for a user to search for the target image. We model the system as a closed queueing network with four queues, depicted in Fig. 5-4. The first queue "stores" the users

Figure 5-4: Model for the system as closed queueing network.

In the following, we derive an expression for the mean response time using the Norton equivalent theorem of queueing networks [13]. The Norton equivalent network with a total of M users is shown in Fig. 5-5.

To find the state-dependent service rate, s_i, we use the approach presented in

Figure 5-5: Norton equivalent network.

[13]. Let

$$\mu_1(i) = \mu_1, \quad \mu_2(i) = i\mu_2/\alpha, \quad \mu_3(i) = i\mu_2/(1-\alpha)$$

$$X_i(k) = \prod_{j=1}^{k} \frac{y_i}{\mu_i(j)}, \quad i = 1, 2, 3, \quad k = 0, 1, ..., M,$$

where $y_1 = y_2 = 1/pq, y_3 = 1/q, y_4 = 1$ is a solution of the balance equation (equation (4) in [13]) and p, q are as defined in Section 5.2.2.

Let

$$\begin{aligned} G_1(k) &= X_1(k) \\ G_2(k) &= \sum_{i=0}^{k} G_1(i)X_2(k-i), \ k = 0, 1, ..., M, \\ G_3(k) &= \sum_{i=0}^{k} G_2(i)X_3(k-i) \end{aligned} \tag{5.3}$$

The state-dependent service rate, s_i, is given by

$$s_i = \frac{G_3(i-1)}{G_3(i)}.$$

If we define the state of the system as the number of requests at buffer B in Fig. 5-5, the state process is a finite population birth-death process with birth-death rates given by

$$\lambda_i = \begin{cases} (M-i)\lambda & 0 \le i \le M-1 \\ 0 & i \ge M \end{cases}$$

and

$$s_i = \frac{G_3(i-1)}{G_3(i)}, \quad 1 \le i \le M,$$

respectively, where $G_3(i)$ is given in (5.3).

The steady-state mean queue size and response time can easily be obtained and are given by [42]:

$$E[q] = P_0 \sum_{i=1}^{M} i \prod_{j=0}^{i-1} \frac{\lambda_j}{s_{j+1}} \tag{5.4}$$

and

$$E[d] = \frac{E[q]}{\lambda(M - E[q])} \tag{5.5}$$

respectively, where

$$P_0 = (1 + \sum_{i=1}^{M} \prod_{j=0}^{i-1} \frac{\lambda_j}{s_{j+1}})^{-1}.$$

The optimal value of α is found by minimizing $E[d]$ over α, $0 \le \alpha \le 1$. The results are summarized in Figs. 5-6, 5-7, 5-8. The most important to note point is that the optimal operating point is not a function of t, the number of users M or μ_2. Fig. 5-6 shows the delay vs. α tradeoff for two values of t. The relative gain of using the α_{opt} is nearly the same in both cases. Fig. 5-7 shows the same tradeoff for different values of μ_2. Note that in the bottom two curves $\mu_2 \ll \mu_1$ and therefore the delay due to the image transmission dominates the delay due to the database access. However, for $\mu_2 = 0.01$ the dominant term is the database delay and little can be gained by choosing a correct α. As was to be expected, optimizing α only makes sense when communication resources are the bottleneck. In Fig. 5-8 the service rate for the transmission is only ten times slower than that of the database access and we can see that when the number of users increases over ten the dominating factor becomes the database access delay, and therefore the choice of α does not make as much of a difference (because the users share the database access but not the communication resources).

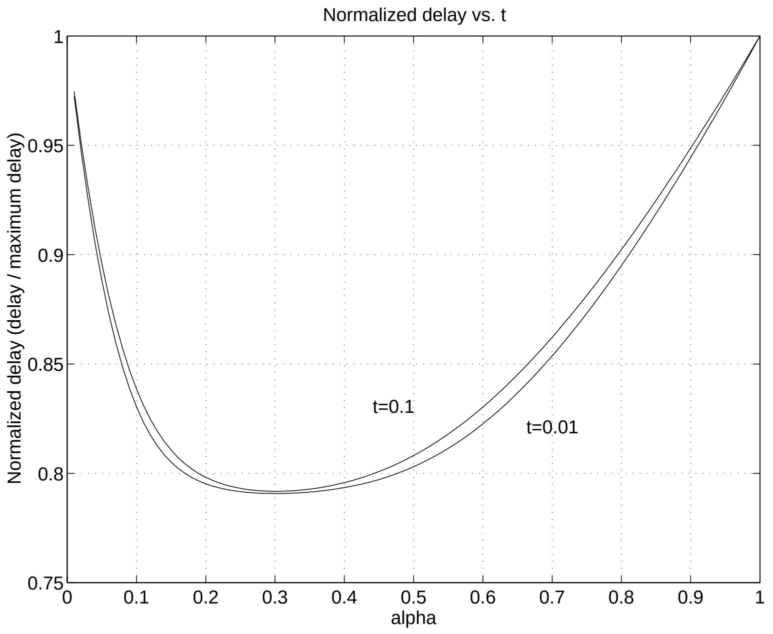

Figure 5-6: Total delay as a function of α for two values of t. In all cases we have that $\alpha_{opt} = 0.3012$. The other parameters are set to $M = 10, m = 5, \mu_1 = 0.1, \mu_2 = 0.01, \lambda = 0.1$. Note that the trade-off is practically identical for both values of t.

5.3.1.2 Separate channels and small image set: non constant t

The results in the previous section indicate that the value of the optimal α does not change with the number of users in the system. In this section, we consider only one user. There are initially N_0 unsearched icons but now we assume that N_0 is "small", so that the probability that one chooses the right icon among i unsearched icons is assumed to be $t(i)$, a function of i. In the following, we will derive an expression for the average delay, $E_\alpha(i)$, incurred in searching the target image, given there are

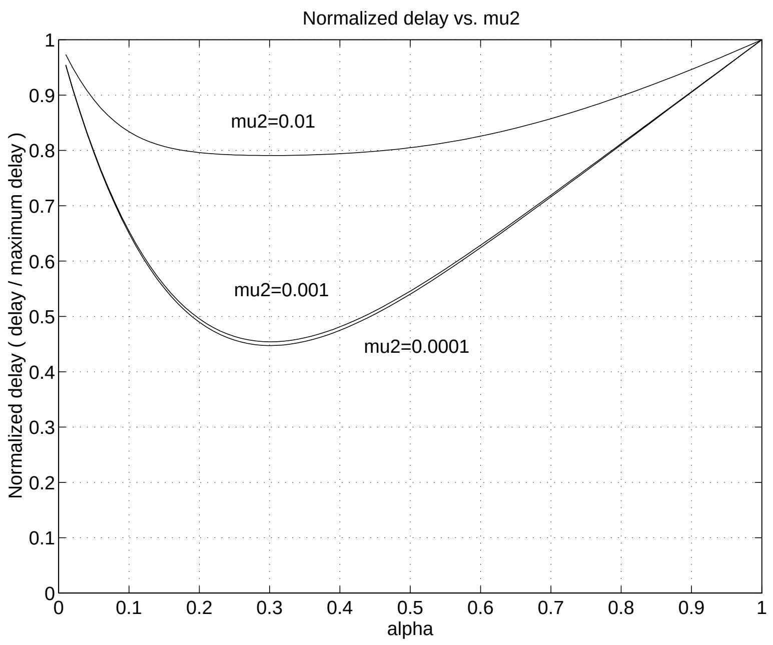

Figure 5-7: Total delay as a function of α for several values of μ_2. In all cases we have that $\alpha_{opt} = 0.3012$. The other parameters are set to $M = 10, m = 5, \mu_1 = 0.1, t = 0.05, \lambda = 0.1$. Note that for $\mu_2 = 0.01$ the delay due to the database access, $\mu_1 = 0.1$, is still significant so that optimizing the transmission results in modest gains. Conversely, for the other two values of μ_2 transmission dominates the delay.

i unsearched icons. Using renewal theory, we have

$$E_\alpha(i) = 1/\mu_1 + \alpha/\mu_2 + (1 - t(i))P(\alpha)E_\alpha(i-1)$$

$$+(t(i)P(\alpha) + (1 - P(\alpha)))((1-\alpha)/\mu_2 + (1 - t(i)/(t(i)P(\alpha) + (1 - P(\alpha))))E_\alpha(i-1))$$

$$= (1 - t(i))E_\alpha(i-1) + \frac{1}{\mu_1} + \frac{1}{\mu_2} - \frac{1}{\mu_2}(1 - t(i))P(\alpha)(1-\alpha) \tag{5.6}$$

and

$$E_\alpha(1) = \frac{1}{\mu_1} + \frac{1}{\mu_2}. \tag{5.7}$$

An explicit expression of $E_\alpha(i)$ can be obtained iteratively from (5.6) and (5.7)

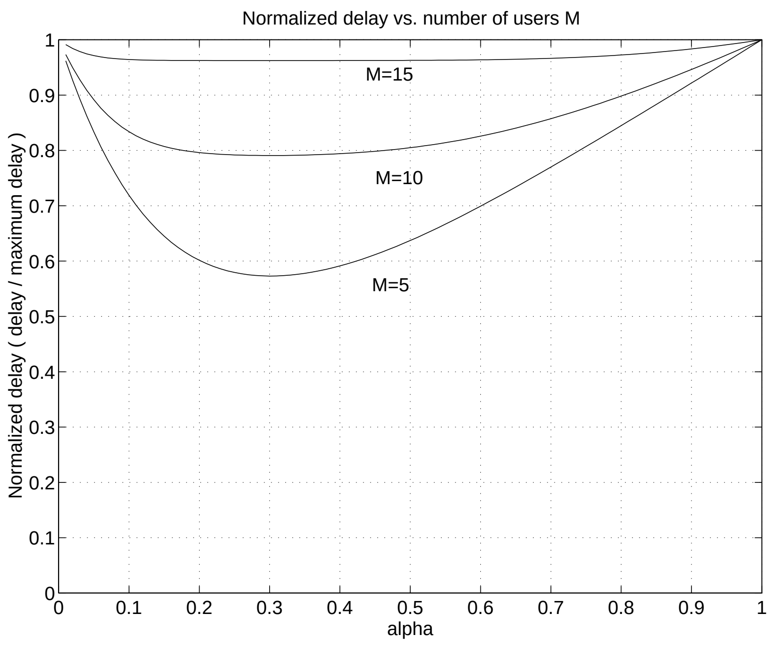

Figure 5-8: Total delay as a function of α for several values of the number of users. In all cases we have that $\alpha_{opt} = 0.3012$. The other parameters are set to $m = 5, \mu_1 = 0.1, \mu_2 = 0.01, t = 0.05, \lambda = 0.1$. Note that, as all users share the database, increases in M imply that the database delay becomes more significant and the gains obtained by optimizing the transmission are smaller.

and is given by

$$E_\alpha(i) = (\frac{1}{\mu_1} + \frac{1}{\mu_2})(1 + \sum_{j=2}^{i} \prod_{k=j}^{i} (1 - t(k))) - \frac{1}{\mu_2} P(\alpha)(1 - \alpha) \sum_{j=2}^{i} \prod_{k=j}^{i} (1 - t(k)) \quad (5.8)$$

Since $t(i) \leq 1$ for all i, $1 \leq i \leq N_0$, we have $\sum_{j=2}^{i} \prod_{k=j}^{i} (1 - t(k)) \geq 0$. Therefore, the problem of minimizing $E_\alpha(i)$ subject to $0 \leq \alpha \leq 1$ is equivalent to the problem of maximizing $P(\alpha)(1 - \alpha)$ subject to $0 \leq \alpha \leq 1$.

So that we have

$$\alpha_{opt} = \arg \max_{0 \leq \alpha \leq 1} (P(\alpha)(1 - \alpha)). \quad (5.9)$$

For the same set of parameters, α_{opt} is exactly the same as that obtained in the previous section (although here our analysis only covers the single-user case).

5.3.1.3 Shared Resources

We now consider the case where all the users share one communication channel. As in Section 5.3.1.1, the system can be modeled as a closed queueing system (see Fig. 5-4), with modified transmission rates which can be determined as follows. The number of users at stages 2 and 3 in Fig. 5-4 represents the number of users sharing the transmission link. At any given time, a user can either be in stage 2 or stage 3, but cannot be in both at the same time. Therefore, since the link is shared, if there are i and j users at stages 2 and 3 respectively, the transmission rates for one user would be $\frac{\mu_2}{(i+j)\alpha}$ and $\frac{\mu_2}{(i+j)(1-\alpha)}$ at stages 2 and 3, respectively. The rate at which at least one user finishes transmitting would be $\frac{i\mu_2}{(i+j)\alpha}$ and $\frac{j\mu_2}{(i+j)(1-\alpha)}$ at stages 2 and 3, respectively.

Because the transmission rate depends on the number of users at other queueing systems, we cannot use the Norton equivalent theorem of queueing networks. Instead, we solve the steady state probability directly. We define the system state as (x_1, x_2, x_3), where x_1, x_2, and x_3 denote the numbers of users at the last three queueing systems in Fig. 5-4, respectively, and $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0,\ x_1 + x_2 + x_3 \leq M$. The one step transition probability, $p(i_1, i_2, i_3 | j_1, j_2, j_3) = p(x_1 = i_1, x_2 = i_2, x_3 = i_3 | x_1 = j_1, x_2 = j_2, x_3 = j_3)$, $0 \leq i_1 + i_2 + i_3 \leq M,\ 0 \leq j_1 + j_2 + j_3 \leq M$, is given as follows:

$$p(i_1 + 1, i_2, i_3 | i_1, i_2, i_3) = (M - i_1 - i_2 - i_3)\lambda$$

$$p(i_1 - 1, i_2 + 1, i_3 | i_1, i_2, i_3) = \mu_1$$

$$p(i_1, i_2 - 1, i_3 + 1 | i_1, i_2, i_3) = p\frac{i_2\alpha}{i_2 + i_3}\mu_2$$

$$p(i_1 + 1, i_2 - 1, i_3 | i_1, i_2, i_3) = (1 - p)\frac{i_2\alpha}{i_2 + i_3}\mu_2$$

$$p(i_1 + 1, i_2, i_3 - 1 | i_1, i_2, i_3) = (1 - q)\frac{i_3(1 - \alpha)}{i_2 + i_3}\mu_2$$

$$p(i_1, i_2, i_3 - 1 | i_1, i_2, i_3) = q \frac{i_3(1-\alpha)}{i_2 + i_3} \mu_2$$

$$p(i_1, i_2, i_3 | i_1, i_2, i_3) =$$

$$1 - (M - i_1 - i_2 - i_3)\lambda - \frac{i_2 \alpha}{i_2 + i_3} \mu_2 - \frac{i_3(1-\alpha)}{i_2 + i_3} \mu_2 - I(i_1)\mu_1$$

$$p(j_1, j_2, j_3 | i_1, i_2, i_3) = 0 \quad \text{for other values of } j_1, j_2, j_3$$

where $I(x)$ is the indicator function, i.e., $I(x) = 1$ if $x > 0$ and $I(x) = 0$ if $x = 0$. The steady state probability $\pi(i_1, i_2, i_3) = p(x_1 = i_1, x_2 = i_2, x_3 = i_3)$ can be obtained by solving the balance equations and the normalized equation, $\sum \pi(i_1, i_2, i_3) = 1$. The total number of states is $M(M^2 + 6M + 11)/6$. The mean delay is given by

$$E[d] = \frac{E[q]}{\lambda(M - E[q])} \tag{5.10}$$

where

$$E[q] = \sum_{l=1}^{M} \sum_{i_1+i_2+i_3=l} l \, \pi(i_1, i_2, i_3).$$

Numerical results for $M \leq 10$ indicate that the optimal value of α is independent of the value of M and once again identical to that obtained in Sections 5.3.1.1, 5.3.1.2. Fig. 5-9 shows the delay vs. α tradeoff for different number of users, while Fig. 5-10 shows the tradeoff when t varies.

5.3.2 Static Analysis

We deemed “dynamic” the analysis introduced above because we set up a model where queries could be terminated at any given time with a certain probability. We now impose the restriction that the user has to examine *all* the candidate images to make a decision. We describe this approach as “static” in the sense that the order in which the images are examined no longer matters, and one can concentrate on the

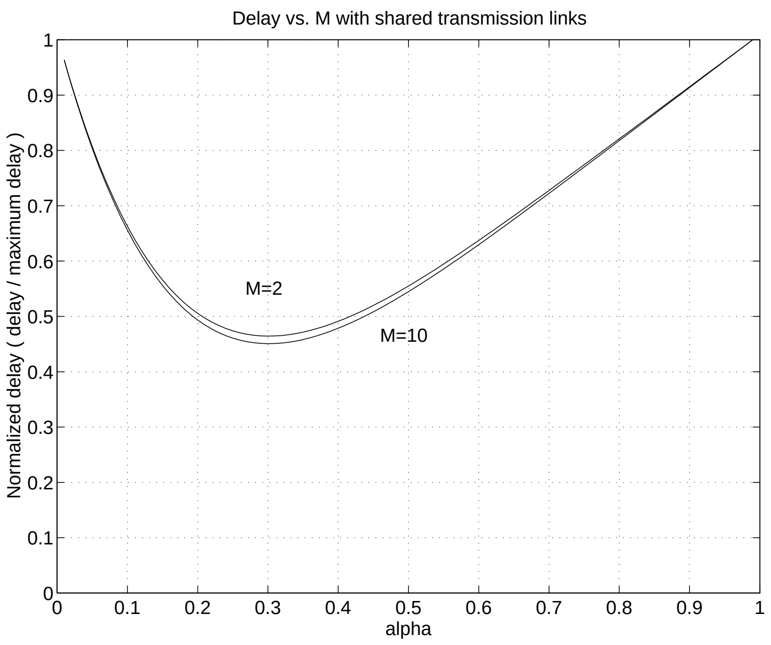

Figure 5-9: Total delay as a function of α for two values of the number of users when the commnunication resources are shared. In all cases we have that $\alpha_{opt} = 0.3012$. The other parameters are set to $m = 5, \mu_1 = 0.1, \mu_2 = 0.01, t = 0.05, \lambda = 0.1$. The relative gain is practically the same regardless of the number of users because the transmission delay dominates as the links are shared.

average behavior.

5.3.2.1 Single subresolution case

The user has now access to N icons simultaneously and can select N_1 icons to be expanded to an intermediate quality level based on the following criterion: if the image quality is sufficient expand the image *only* if it is one of the targets, else expand regardless. Out of the N_1 intermediate resolution images again N_2 are selected to be displayed at full resolution. The same criterion is applied at this step. Note that the main difference between this model and the previous one is that here the user selects images in batches, whereas before a sequential selection was performed. Thus we are in effect modeling the user interaction "with memory" mentioned in Section 5.2.1. Suppose that we know there are n images in the original set of icons that can be

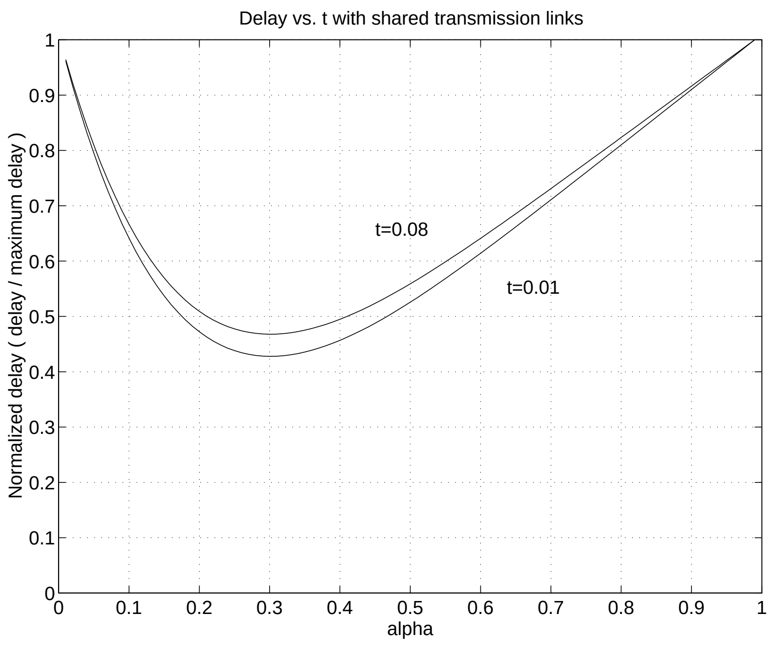

Figure 5-10: Total delay as a function of α for two values of t when the communication resources are shared. In all cases we have that $\alpha_{opt} = 0.3012$. The other parameters are set to $m = 5, \mu_1 = 0.1, \mu_2 = 0.01, M = 5, \lambda = 0.1$

of interest to the user, then at every step at least n images are selected while in addition some images are selected because quality was not sufficient to decide. The user will make a final choice among the N_2 remaining images but since all of them have already been downloaded we assume there is no cost involved and we will ignore this step in our model.

Assume that, if B is the number of bits for the original images, then αB bits are used for the intermediate resolution images (where $0 \leq \alpha \leq 1$). Even though the images have different sizes, in order to make the selection the user will have to go through all of them so that the optimal cost depends only on the relative sizes of the subresolution images, i.e. α, and not on the actual sizes of the images. Therefore, to simplify our analysis we can assume all images have the same size. Note that this is consistent with the results of the dynamic analysis where the α_{opt} did not change with μ_2. Then, using the same $P(\alpha)$ as before, our aim is to minimize the total

m	α
2	0.4227
3	0.3700
4	0.3313
5	0.3012

Table 5.1: Optimal α for several expressions of $P(\alpha)$. Solutions obtained using the static analysis.

transmission delay:

$$J = N_2 B + (N_1 - N_2)\alpha B \tag{5.11}$$

where we just have added the bit rates of the images at each resolution level. Obviously the quantity to minimize is

$$J' = N_2 + (N_1 - N_2)\alpha \;=\; (1-\alpha)N_2 + \alpha N_1. \tag{5.12}$$

Now assuming the number of images is sufficiently large, we have that on average:

$$N_2 = P(\alpha)n + (1 - P(\alpha))N_1. \tag{5.13}$$

Using (5.13) in (5.12) we get:

$$J' = N_1 + (n - N_1)P(\alpha)(1-\alpha). \tag{5.14}$$

Therefore, since $n \leq N_1$ our objective in order to minimize the cost is to maximize $P(\alpha)(1-\alpha)$, as we had already derived in Section 5.3.1.2. The solutions obtained for $P(\alpha) = 1 - (1-\alpha)^m$ for $m = 2, \cdots, 5$ are summarized in Table 5.1. The results match those obtained following the dynamic analyses of Section 5.3.1.

m	α_1	α_0	Gain (%)
2	0.4227	0.2803	35.6
3	0.3700	0.2271	44.2
4	0.3313	0.1924	50.8
5	0.3012	0.1676	56.1

Table 5.2: Optimal α_1 and α_0 for several expressions of $P(\alpha)$. Solutions obtained using the static analysis in the two resolution case. The gain column indicates the reduction in total delay obtained when going from one to two resolutions.

5.3.2.2 Multiple layers

Under the same assumptions of a large image set and a search through all the images in the set, we can extend the above analysis to the case where there are multiple layers of resolution.

As an example, consider the case where there are two layers: the lower resolution layer uses $\alpha_0 B$ bits per image, while the next higher layer uses $\alpha_1 B$ bits. Then following a procedure analogous to that just outlined it can be seen that the optimal pair $(\alpha_0, \alpha_1)_{opt}$ is such that:

$$(\alpha_1, \alpha_0)_{opt} = \arg \max_{(\alpha_1,\alpha_0)} \left(\ (1-\alpha_1)P(\alpha_1)(1-P(\alpha_0)) + (1-\alpha_0)P(\alpha_0) \ \right) \qquad (5.15)$$

The results are summarized in Table 5.2. Note how using two intermediate resolutions instead of one produces a significant reduction in the overall delay. Also note that the optimal allocation for the higher resolution α_1 is the same as obtained in the one-layer case. This, as will be seen shortly, is an indication that the problem can be solved iteratively.

In the more general case, the images are accessed at k resolutions, such that $\alpha_{k-1}, \ldots, \alpha_1, \alpha_0$, are the relative sizes of the images, with $\alpha_{k-1} \geq \alpha_{k-2} \geq \ldots \geq \alpha_1 \geq \alpha_0$. Using an analogous notation, we will have initially N icons, then N_0 images which use $\alpha_0 B$ bits, N_1 that use $\alpha_1 B$ bits and so on. At the last stage there will be N_k images displayed at full resolution. Under the same assumptions as before we can write:

$$N_1 = P(\alpha_0)n + (1 - P(\alpha_0))N_0, \quad N_2 = P(\alpha_1)n + (1 - P(\alpha_1))N_1, \quad \ldots$$

and so on, up to

$$N_k = P(\alpha_{k-1})n + (1 - P(\alpha_{k-1}))N_{k-1}.$$

Then, the cost to minimize is J_k, with

$$J_k = N_k + \alpha_{k-1}(N_{k-1} - N_k) + \ldots + \alpha_0(N_0 - N_1) = N_k + \sum_{j=0}^{k-1} \alpha_j (N_j - N_{j+1}) \quad (5.16)$$

From the recursive definition of the N_i above, we can see that:

$$N_{k-1} - N_k = P(\alpha_{k-1})(N_{k-1} - n) = P(\alpha_{k-1})(N_{k-1} - N_{k-2} + N_{k-2} - n)$$

so that

$$\begin{aligned} N_{k-1} - N_k &= P(\alpha_{k-1})(-P(\alpha_{k-2})(N_{k-2} - n) + N_{k-2} - n) \\ &= P(\alpha_{k-1})(1 - P(\alpha_{k-2}))(N_{k-2} - n) \end{aligned}$$

and in general, using repeatedly the same method,

$$N_{k-1} - N_k = P(\alpha_{k-1}) \left[\prod_{j=0}^{k-2} (1 - P(\alpha_j)) \right] (N_0 - n). \quad (5.17)$$

The cost can now be rewritten, by adding and subtracting N_j, $j = k-1, \cdots, 0$ to

(5.16), as

$$J_k = N_0 + \sum_{j=0}^{k-1} (\alpha_j - 1)(N_j - N_{j+1}) \tag{5.18}$$

so that using (5.17) we can write:

$$J_k = N_0 + \left\{ \sum_{j=0}^{k-1} (1 - \alpha_j) P(\alpha_j) \left[\prod_{l=0}^{j-1} (1 - P(\alpha_l)) \right] \right\} (n - N_0) \tag{5.19}$$

and therefore, since $n - N_0 < 0$, the solution to our problem is:

$$(\alpha_{k-1}, \ldots, \alpha_1, \alpha_0)_{opt} = \arg \max_{(\alpha_{k-1}, \ldots, \alpha_1, \alpha_0)} \left\{ \sum_{j=0}^{k-1} (1 - \alpha_j) P(\alpha_j) \left[\prod_{l=0}^{j-1} (1 - P(\alpha_l)) \right] \right\}, \tag{5.20}$$

where the quantity to maximize will be denoted C_{k-1} and we have the constraints that

$$0 \leq \alpha_0 \leq \alpha_1 \leq \ldots \leq \alpha_{k-1} \leq 1. \tag{5.21}$$

This is a multidimensional optimization problem, which due to the particular structure of the objective function can be solved iteratively, so that at each iteration only a one-dimensional problem has to be solved.

To simplify the notation let us define

$$\Pi_i = \prod_{j=0}^{i} (1 - P(\alpha_j)).$$

Then we can write

$$C_{k-1} = (1 - \alpha_{k-1}) P(\alpha_{k-1}) \Pi_{k-2} + C_{k-2}$$

where it is clear that only the term $F_{k-1}(\alpha_{k-1}) = (1 - \alpha_{k-1}) P(\alpha_{k-1})$ depends on α_{k-1}.

The iterative solution is as follows. We first choose

$$\alpha^*_{k-1} = \arg\max_{\alpha_{k-1}} (1-\alpha_{k-1})P(\alpha_{k-1})$$

and let $F^*_{k-1} = F_{k-1}(\alpha^*_{k-1})$. Our objective function C_{k-1} can now be written as

$$C_{k-1} = (F^*_{k-1} \cdot (1 - P(\alpha_{k-2})) + (1-\alpha_{k-3})P(\alpha_{k-3}))\Pi_{k-3} + C_{k-3}$$

and we choose next α^*_{k-2} as

$$\alpha^*_{k-2} = \arg\max_{\alpha_{k-2}} (F^*_{k-1} \cdot (1 - P(\alpha_{k-2})) + (1-\alpha_{k-2})P(\alpha_{k-2}))$$

where we again just consider the terms that depend on α_{k-2}. At every stage j we have thus

$$C_{k-1}(\alpha^*_{k-1}, \cdots, \alpha^*_{j+1}) = (F^*_{j+1} \cdot (1 - P(\alpha_j)) \; + \; (1-\alpha_j)P(\alpha_j))\Pi_{j-1} \; + \; C_{j-1}$$

and we choose α^*_j to maximize

$$F_j(\alpha) = (F^*_{j+1} \cdot (1 - P(\alpha)) + (1-\alpha)P(\alpha))$$

where F^*_{j+1} represents the maximum obtained in the previous iteration. The procedure is iterated until α^*_0 is obtained.

The terms on the different α_j's are independent of each other and are all positive so that maximizing the terms separately also yields the overall maximum. Therefore the solution we obtain through the iterative procedure provides the maximum of C_{k-1} for all k-tuples, $(\alpha_{k-1}, \ldots, \alpha_0)$, such that $0 \leq \alpha_i \leq 1$. However, the solution we are seeking requires that, additionally, the constraint of (5.21) is met. We now verify that the iterative solution *also meets this additional constraint* under the sole requirement that $P(\alpha)$ be non-decreasing in α, with $P(0) = 0$ and $P(1) = 1$ (which is a reasonable assumption: the more bits we use the better chances we have of getting sufficient quality).

Consider the j-th stage of the iteration, we have

$$F_j^* = \max_\alpha (F_{j+1}^* \cdot (1 - P(\alpha)) + (1 - \alpha)P(\alpha))$$

and

$$F_{j-1}^* = \max_\alpha (F_j^* \cdot (1 - P(\alpha)) + (1 - \alpha)P(\alpha)).$$

Clearly, since $F_j(\alpha = 0) = F_{j+1}^*$, and $F_j^* = \max_\alpha F_j$, we have $F_j^* \geq F_{j+1}^*$. Assume for instance $F_j^* = F_{j+1}^* + \delta$, with $\delta \geq 0$. Then

$$\alpha_{j-1}^* = \arg\max_\alpha \left\{ (F_{j+1}^* + \delta)(1 - P(\alpha)) \;+\; P(\alpha)(1 - \alpha) \right\}$$

so that

$$\alpha_{j-1}^* = \arg\max_\alpha \left\{ \delta(1 - P(\alpha)) \;+\; F_{j+1}^*(1 - P(\alpha)) \;+\; P(\alpha)(1 - \alpha) \right\}$$

The first term cannot increase if α increases (because $P(.)$ is non-decreasing), while the last two terms have a maximum, by definition, at $\alpha = \alpha_j^*$. Therefore, since both terms decrease for $\alpha \geq \alpha_j$ we must have that:

$$\alpha_{j-1}^* \leq \alpha_j^*. \tag{5.22}$$

Thus, we have verified that the iterative solution, which provides the optimal solution under the constraints $0 \leq \alpha_i \leq 1$, produces a solution that *also* meets the constraint of (5.21). Therefore, we indeed have that $(\alpha^*_{k-1}, \ldots, \alpha^*_0)$ is the solution to (5.20) with constraint (5.21). Fig. 5-11 shows how the overall delay can be reduced by adding intermediate resolution layers to the system. As was to be expected, increasing the number of layers reduces the overall delay at the cost, however, of increased complexity.

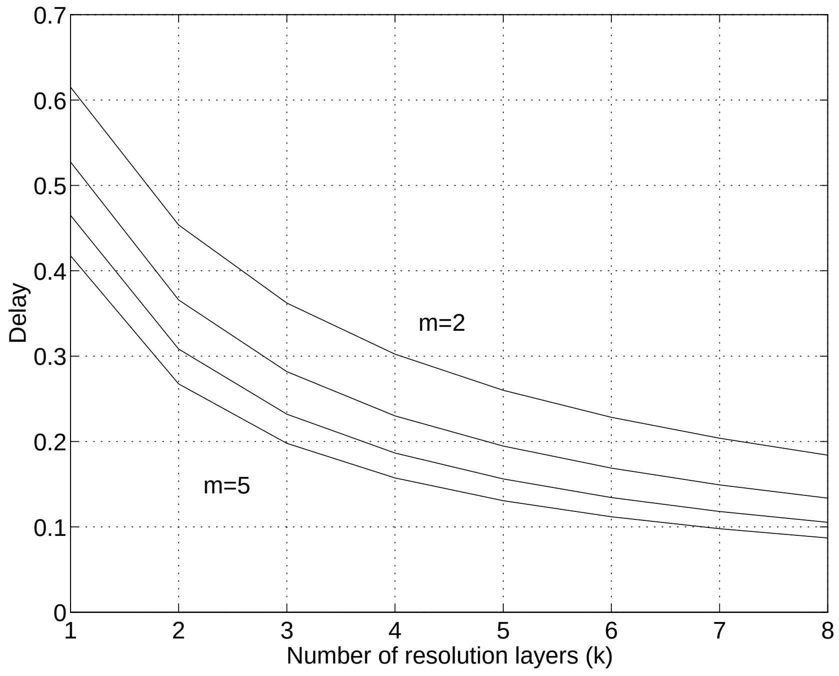

Figure 5-11: Overall delay at the optimal combination of α's for different values of m. Note how the delay can be significantly decreased by increasing the number of resolutions, at the cost of increased complexity.

5.3.3 Discussion

A first conclusion of the foregoing sections is that finding the optimal operating point can be worthwhile in reducing the overall delay, in particular in cases where users are connected to the database through low-speed links. For instance choosing the optimal α can provide reductions in delay of up to a factor of two in the $m = 5$ case (see Fig. 5-12 in the static analysis or Figs. 5-9-5-10 for the dynamic analysis in the shared resources case). Moreover, note how the advantage of choosing a correct value for α increases as the parameter m, which determines the shape of $P(\alpha)$, increases. In most cases of interest one can expect a relatively large m to be likely, i.e. a relatively small percentage of the total bit rate provides sufficient quality to make a decision.

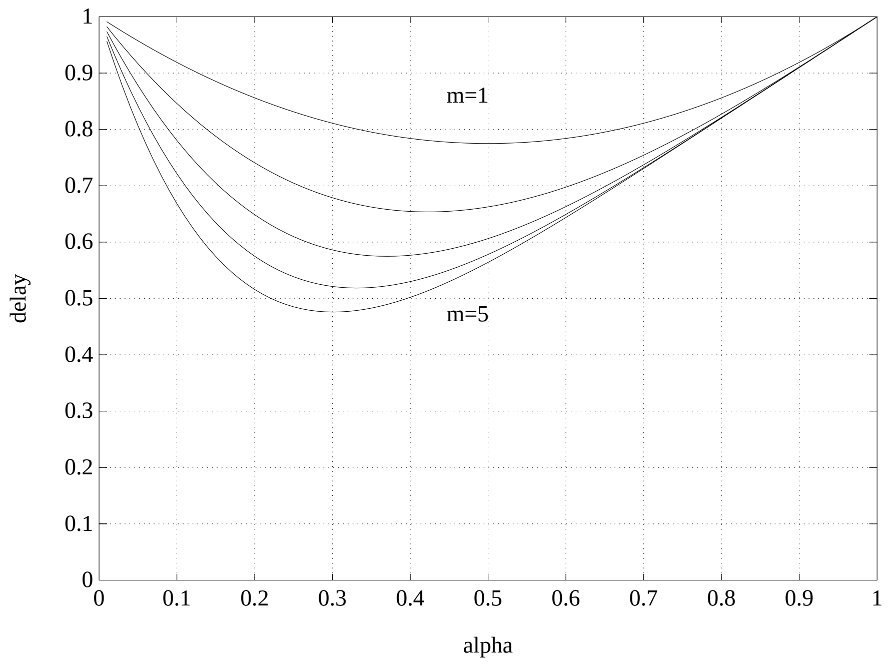

Figure 5-12: Example of total normalized delay for the range of possible values of α for several values of the parameter m in the expression for $P(\alpha)$. Note how choosing the right value for α can reduce significantly the total delay. This example models the cost as in (5.14) where we assume $n \ll N_1$ so that the delay is approximately $1 - P(\alpha)(1 - \alpha)$

A second point to note is that the optimal α is independent of the exact procedure that is used for transmission. For instance, the static and dynamic analyses provide identical results even though they assume different ways of proceeding with the browsing. Similarly we find the same results for α whether one or several users access the database, and whether or not the users share the transmission resources. Finally, we see no dependence of the optimal result on the size of the initial image set, or the probability of getting the correct image, t (equivalently, in the static analysis, the number of correct images n).

The intuitive justification is that the exact procedure for retrieving the images is not relevant because we are concerned with minimizing an average cost. Since for every image we have an average measure of the "risk" of having to retrieve the rest of the image (i.e. $P(\alpha)$) and we assume all images are identical (i.e. same probability) it is normal to expect that the only factor to determine the optimal operating point would be $P(\alpha)$.

Similarly, as we increase the number of users, and even if the transmission resources are shared, the optimal value for α remains unchanged. This is again due to our choosing to minimize the average delay for a set of users that are identical, at least in a statistical sense.

Even though the optimal operating point is independent of the system parameters, the gain of using a multiresolution approach is not. In particular we pointed how more gain can be expected when the transmission resources are shared or the transmission resources, rather than the database, represent the bottleneck of the system.

Finally, it should be mentioned that our analysis of the multiple resolution layers case indicates that substantial reductions in delay are possible by using more than one intermediate resolution. Systems with several intermediate resolutions should thus be considered provided that the increase in implementation complexity can be afforded.

5.4 Conclusions and Future Work

In this chapter, we have addressed a problem that arises when designing a remote image retrieval system, namely, that of assigning bits to the different layers of the images to be transmitted. We have solved this problem under assumptions for the average quality of the images ($P(\alpha)$) and the restriction that all the images use the same bit allocation. Results show that significant gain can be expected from choosing a correct bit allocation quite independently of the exact procedure that is used to retrieve the images.

Our analysis leaves a number of questions for future work. In particular it would be of interest to perform quality measures on real images to obtain empirical expressions for $P(\alpha)$. Also, since the average analysis provides the same results as the dynamic one, it would be interesting to relax the constraint on the bit allocation and allow each image to have a different α. Thus each image would have its own probability of having sufficient quality and thus we could setup a "static" bit allocation problem among the images: to give more bits to those images where the bits can do more "good", in the sense of reducing the overall delay.