Preloading policies for the Virtual Computing Lab

KEERTHANA BOLOOR
ECE Department, North Carolina State University
and
AARON PEELER
VCL, North Carolina State University
and
JOSH THOMPSON
VCL, North Carolina State University
and
YANNIS VINIOTIS
ECE Department, North Carolina State University

In the Virtual Computing Laboratory (VCL) users request a computing resource (e.g., a blade) and customized software (an "image"). The software is loaded onto the blade from an image server. If loading is performed at the time of the request, the user experiences a delay before s/he can access the blade. This delay may be several minutes long, depending on the type of image requested. A "preloading policy" aims at reducing this delay; the basic idea is to anticipate what images may be requested in the future and start loading them onto free blades in advance of reservations. In this paper, we define several preloading policies and show via simulation that they can significantly reduce user-experienced delays. We drive the simulation with two sets of workloads: (a) actual traces derived from historical data collected over several years of operations at North Carolina State University's VCL (approximately 155,000 reservations), and, (b) synthetic workloads. Performance metrics include user-related ones (e.g., waiting time histograms and averages) and system-related ones (e.g., blade/image utilization). We use the simulator to address "what if", system design questions and select system configuration parameters.

Categories and Subject Descriptors: C.4 [**PERFORMANCE OF SYSTEMS**]: Performance attributes

General Terms: Algorithms, Performance, Experimentation

Additional Key Words and Phrases: Management

The work of K. Boloor and Y. Viniotis was supported in part by a Center for Advanced Computing and Communications (CACC) grant. Corresponding author's address: Department of Electrical and Computer Engineering, Box 7911, NCSU, Raleigh, NC 27695. Email: candice@ncsu.edu.

1. INTRODUCTION AND MOTIVATION

In a Virtual Computing Lab (VCL), users request a computing resource (e.g., a blade) and customized software (an "image", typically an operating system and bundles of applications). The software is loaded onto the blade from an image server. If loading is performed at the time of the request, the user experiences a delay before s/he can access the blade. This delay may be several minutes long, depending on the type of image requested.

A "preloading policy" aims at reducing this delay; the basic idea is to anticipate what images may be requested in the future and start loading them onto free blades in advance of reservations.

The remainder of the paper is organized as follows. In Section 2, we define the preloading policies we have evaluated. In Section 3, we describe the VCL testbed we have used in this study. In Section 4, we evaluate the preloading policies using actual reservation data from approximately four years' worth of VCL operations at NCSU. In Section 5, we evaluate the policies using synthetic workloads and examine how system configuration parameters can be selected to optimize performance criteria.

2. EXAMPLES OF PRELOADING POLICIES

Let's define

—N: number of blades in the system. All blades are assumed equivalent, for simplicity.

—M: number of different types of images to be preloaded.

—$S(I)$: the average amount of time (in minutes) it takes the VCL managing software to load image I onto a free blade.

When the system initializes (e.g., beginning of a new day), the preloading policy is any rule that selects $K(I)$ blades on which to preload image I. The sum

$$\sum_I K(I) = N' <= N.$$

We allow for the sum to be strictly less than N for generality and to save preloading overhead (in case we have "networking/power consumption" constraints). N' is a configurable constant.

Some examples of initial preloading policies (IPP) include:

—**IPP1**: Uniform, maximal preloading. Set

$$K(I) = floor(N'/M),\ for\ all\ I.$$

—**IPP2**: Uniform, minimal preloading. Set

$$K(I) = 1,\ for\ all\ I.$$

While the system is operational, the operational preloading policy (OPP) is any rule that determines what image to load on a free blade. We define a blade as free if it is turned on, not assigned to a user and not preloading any image. We need some data structures in support of the OPP. Let's define

—$LF(I, t)$: the number of free blades at time t that have already image I loaded,

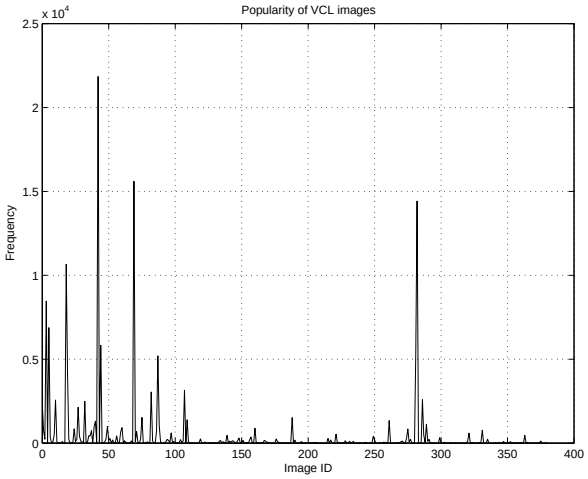

Fig. 1. Popularity of the images in VCL, years 2004-2008.

—$LB(I,t)$: the number of busy blades at time t that have already image I loaded,

—$FR(I)$: the "popularity" (e.g., the frequency of use) of image I.

Figure 1 depicts the popularity of the images in VCL.

The decision instant t can be the arrival time of a reservation request, the time when a user relinquishes a blade, or any policy-dictated instant; for example, the decision instants can form a periodic sequence. When a blade becomes free, at time t, we assume that the existing image is "erased" since in most cases the users have administrative privileges and hence cannot be trusted.

The preloading policies can be based on the data structures we defined. One basic idea is to try to have, at any time instant, at least one copy of an image loaded, on a free blade, so that the incoming instantaneous reservation request can be granted without the user incurring the $S(I)$ waiting time.

Some examples of operational preloading policies (OPP) we have evaluated are:

OPP1: Uniform, maximal set of available, free images. Sort $LF(I,t)$ with respect to I, in increasing order. Load the image I that corresponds to the minimum $\{LF(I,t)\}$. Break ties randomly.

OPP2: Uniform, maximal set of available, free + busy images. Sort $LF(I,t) + LB(I,t)$ with respect to I, in increasing order. Load the image that corresponds to the minimum $\{LF(I,t) + LB(I,t)\}$. Break ties randomly.

OPP3: Most frequent image preloading. Sort $FR(I)$ in decreasing order. Sort $LF(I,t)$ with respect to I, in increasing order. Load the image that corresponds to the minimum $\{LF(I,t)\}$. Break ties according to $FR(I)$ order.

OPP4: Threshold-limited preloading. OPP1-OPP3 can be thought of as attempts to maximize user-centric performance measures. OPP4 aims to minimize preloading activities (power consumption) when the administrator deems that there are "enough" preloaded images of any type. OPP4 makes sure that the number of preloaded copies of an image does not exceed a given, configurable threshold; for

example, it makes sure that

$$LF(I,t) <= K, \quad for\ all\ I.$$

(K is a configurable constant; $K = 2$, for example).

The definition of OPP1-OPP4 is based on the following principles: (a) they should make use of historical data, (b) they should be simple to implement, and (c) they should incur low overhead. All policies use data structures that require $O(M)$ memory, a simple sorting operation on the data structures and historical data ($FR(I)$) which needs to be processed only once. Note that more "sophisticated" processing of historical data is possible (e.g., producing image popularity based on time of day); however, we chose not to experiment with it in the present phase.

3. EXPERIMENTAL ENVIRONMENT AND PERFORMANCE METRICS

3.1 Current VCL environment and historical data

We have historical records from the operation of the NCSU VCL over the years 2004-2008. The records include four main components: arrival times for reservation requests, image requested, duration of request and waiting time for image loading.

Approximately 155,000 records have been collected and used in this study. There are two types of reservation requests which we have excluded from the data for the purposes of this study - future reservations and block reservations (they represented approximately 5% of the total number of requests. Around 35 to 40 minutes before the start time, the machine will start to be preloaded with that reservation's image (if it is idle and not already loaded with it). Block reservations are used primarily for classroom use of VCL. For these, some number of machines is set aside for a specific set of users, for a specific time, using a specific image. The machines are preloaded with the image to be ready at the beginning of the specified time period.

The number of blades, N, has been increasing through the years; currently, $N \approx 150$. The number of images, M, has also been increasing; currently, $M \approx 380$, with about 50 images being used 90% of the time. In Figure 1, we depict the frequency with which images are requested by the users.

In Figure 2, we depict the average waiting time a user spends before their requested image is loaded. Data for such waiting times were recorded only for years 2006-2008 (approximately 40,000 records). The overall system average was found to be around 12 minutes. This is the "benchmark" figure that any preloading policy should improve upon.

3.2 Performance metrics

Performance metrics of interest in this study include user- and system-related statistics:

We define the following waiting time metrics for user requests:

—EW: average waiting time for all users (single number, measured in minutes)

—$EW(I)$: average waiting time per image I (M numbers, measured in minutes)

We define the following system utilization:

—EU: average system utilization (single number, dimensionless)

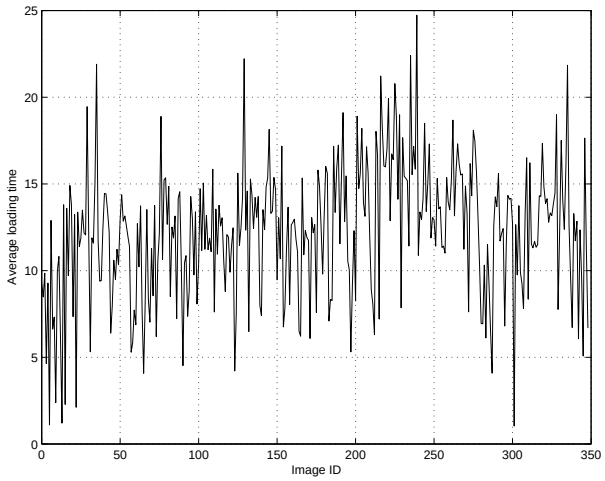

Fig. 2. Average waiting time, without any preloading, as a function of the requested image.

—$EU(I)$: average system utilization per image I (M numbers, dimensionless)

Note that system utilization is directly proportional to power consumption and can be used to define preload policies that aim at reducing it.

4. PRELOADING POLICY ANALYSIS, VCL HISTORICAL DATA

We have evaluated user waiting times, EW and $EW(I)$, blade and system utilization, EU and $EU(I)$, for OPP1-OPP4, using the reservation records from the NCSU VCL.

During part of its operation, VCL has applied some form of limited preloading: Linux images have always been preloaded on a subset of blades. Unlike other images, which are always "removed" from a blade after a reservation is finished, Linux images always remain loaded on the blade. From the historical data, we have calculated that the overall system utilization was quite low; assuming operations 24 hours a day, 7 days a week, the calculated utilization was below 10%.

In figure 3 we plot the EW metric, as a function of the system utilization, EU. N; we have varied the number of blades, N, in order to change EU. In doing so, we have introduced some blocking, which was comparable for all policies. In obtaining the results, we have excluded from the data the preloaded Linux images. Note that OPP3 performs the best in all utilizations, reducing waiting times by about 50% on average (calculated across all utilization levels).

5. PRELOADING POLICY ANALYSIS, SYNTHETIC WORKLOAD DATA

We have then simulated arrival times for reservation requests, image requested, duration of request and waiting time for image loading to analyze "system design, what-if" questions about the effects of system configuration parameters and settings on variations of OPP1-OPP4. For simplicity, arrival times were assumed to be a Poisson process, with rate R requests per minute. The rate R was tuned

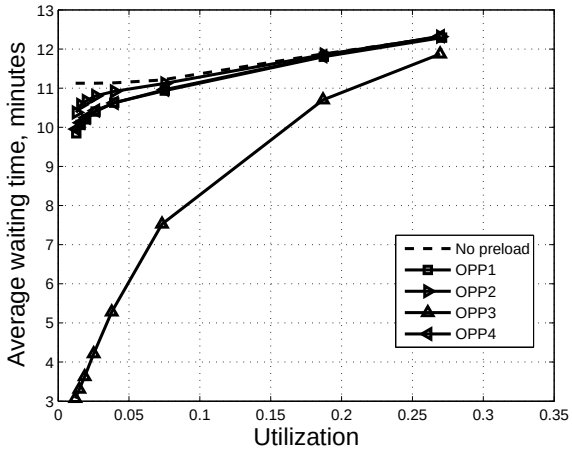

Fig. 3. Average waiting times, OPP1-OPP4 and "no preload" policies.

(along with duration of request) to derive a desired system utilization. The image type was modelled as a random variable, I, drawn from a distribution that matches the popularity of images in the current VCL, 50 most popular images only. The duration of a request was modelled as a random variable, D, drawn from a distribution that matches the current VCL records, including extension times. Finally, the waiting times for image loading were modelled as a random variable, W, drawn from a distribution that matches the current VCL records, 50 most popular images only.

The questions we focused on were:

—How preload policies behave as system utilization increases?

—Can we gain by "designing the number of images" M?

—Is K an "effective parameter for controlling power consumption and user delay?

5.1 System utilization experiments

In Figure 4, we show the behavior of OPP1-OPP4 as system utilization increases. We varied the request arrival rate in this experiment; the blocking probability was comparable for all policies. OPP3 performed best in this experiment as well.

5.2 Number of images experiments

Recall that the analysis of the historical data showed that three out of four preload policies behaved approximately the same. We attribute this behavior mostly to the fact that utilizations were low and the number of images was too high, for these preloading policies to have any significant effect.

In Figure 5, we show the behavior of OPP1 and OPP3 as the number of images is varied. We use a "no preload" policy as a benchmark; the number of blades was set to 155. We have assumed that EL, the average loading time per image grows linearly as a function of the number of supported images, M, with a minimum of 12 minutes (for $M = 1$) and a maximum of 45 minutes (for $M = 50$).

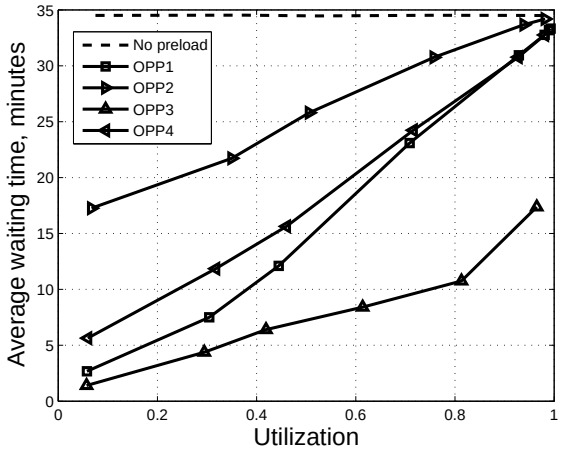

Fig. 4. Average waiting times, OPP1-OPP4 and "no preload" policies, as a function of system utilization.

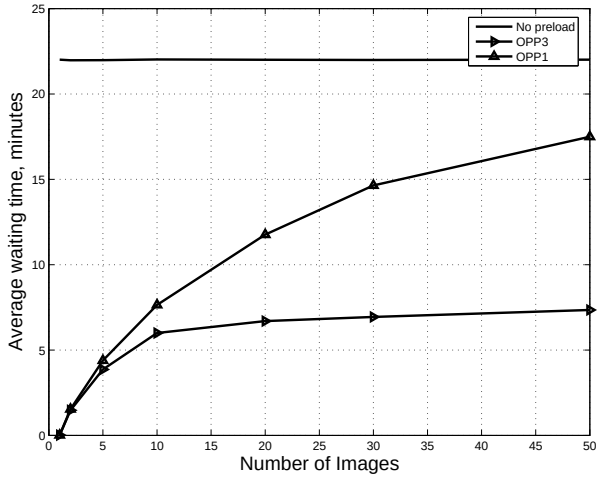

Fig. 5. Average waiting times vs the number of images, M.

Recall that OPP3 takes into account the popularity of images for its preloading decision, while OPP1 does not. As Figure 5 depicts, when the number of images is less than the number of available blades, taking popularity of images into account does have a significant effect. Even a random selection of what image to preload reduces the average waiting time significantly, compared to a no preload policy.

5.3 Effect of the parameter K

In Figure 6, we show the behavior of OPP4 as the parameter K is varied. In this experiment, we have assumed 155 blades and 50 images. The average loading time

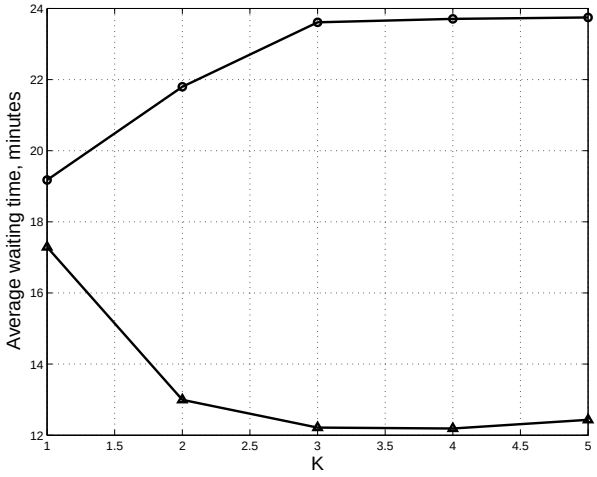

Fig. 6. Average waiting times vs K.

is 45 minutes. Recall that K is a parameter that limits the number of blades with a preloaded image. Setting $K = 2$, for example, would mean that up to $2 \times 50 = 100$ blades would be used for preloading at any time, leaving room for reducing power.

In Figure 6, the curve with circles depicts the behavior of OPP4 with ties broken randomly. The curve with triangles depicts the behavior when the popularity of images is taken into account. We observe that OPP4 reduces the waiting time by a factor of 2 to 4. The effect of the parameter K appears to diminish, as K increases, in both cases. Note that waiting times increase as K increases due to the fact that popularity of images is highly uneven in this experiment. The conclusion of this experiment is that having more than one preloaded image does not really improve waiting times and hence small values of K should be preferred for reducing power costs.

6. CONCLUSIONS AND FURTHER RESEARCH

In this paper, we have defined several preloading policies for use in a VCL environment. We have shown via actual and simulated data that they can significantly reduce user-experienced delays, while at the same time they can also be used to control system utilization/power consumption. The policies are simple to implement and incur limited CPU and memory overhead. Further work can be directed towards (a) defining rules for selecting appropriate image bundles, and, (b) defining "best practices" for setting system configuration parameters.