Energy and Performance: Two Win-Win Examples

Yitzhak Birk
Technion
Haifa 32000, Israel
birk@ee.technion.ac.il

Tomer Kol
Google Inc.*
Haifa, Israel
tomer.kol@google.com

1. BACKGOUND

Energy efficiency and performance often present design a trade-off. However, this is not always so. We present two examples of work that was originally motivated by performance but produced energy savings. One deals with increasing disk-drive throughput in web servers and similar settings, and demonstrates a pure win-win situation. The other deals with multi-channel ALOHA networks used for transaction processing. Here, while there is an apparent trade-off among energy, throughput and response time, careful examination of the performance measures enabled a (win, win, no-lose) solution. The latter example provides a general lesson pertaining to the potential value of addressing the exact requirements rather than ones that "contain" them.

2. "Packaging" web pages for efficient access

2.1 Background

We consider web servers that store mostly static web pages. Our focus is moreover on settings wherein caching a substantial fraction of the data in the web server's memory is too expensive, and caching a small fraction is of little benefit. Such are particularly applicable to hosting of numerous lightly accessed web sites on shared servers, virtualized or otherwise.

Disk drives have improved over the years in terms of both storage capacity, transfer rate and access latency. However, while capacity has improved by 5-6 orders of magnitude over the past 25 years and transfer rates have improved by some 3 orders of magnitude (approximately the square root of the capacity improvement), the number of independent I/O operations per second has become the critical performance measure for disk drives in most I/O-intensive applications.

A web server is measured on its throughput, subject to an acceptable response time. Whenever the disk drives are the limiting factor, it is desirable to minimize their required data access rate per given workload. Stated differently, it is desirable to minimize the amount of *disk work* per given mission, as so doing increases the number of missions that the disk can perform per unit time, otherwise known as its throughput.

Remarks.

1) The term "mission" is used intentionally, as it refers to an amount of work as seen by the user. This should be contrasted with the number of disk accesses, which represents the amount of work done by the disk. (Caching, for example, reduces the latter without changing the former.)

2) Disk work is measured in units of time. However, it is the net time spent by the disk doing work (seek, rotational latency and data transfer) rather than the response time or latency seen by the user or job, as the latter also includes queuing delay. This extremely important measure is the equivalent of "person hours" used when billing for work done by the hour.

A web page is usually organized as a small "skeleton" html file, which normally contains the page's textual information along with calls to embedded objects (typically very small images) and positioning information for each such object. Upon receipt of a request for a web page, the server reads its skeleton file from disk and sends it to the user's browser. The latter parses the skeleton and issues a request for each of the embedded objects. Delivering a single web page to the browser may thus entail more than ten disk accesses by the server. With individual embedded objects being small, the amount of disk work is nearly proportional to the number of these objects.

Two well-known techniques come to mind in order to mitigate the problem: 1) prefetching and 2) proximal placement of the embedded objects on disk. Unfortunately, however, neither of these approaches fully solves the problem. Prefetching is merely a scheduling technique, and does not directly reduce the required amount of disk work. (One could argue that there is some benefit if there are multiple pending requests and the disk drive is allowed to schedule them so as to minimize total seek time.) Similarly, proximal placement alone does not solve the problem: if two objects are stored contiguously on the same track but the disk is not aware of their both being requested, rotational latency will not be reduced; moreover, in a server setting consecutive requests from a given client may be separated by requests from other clients, which may cause a seek to a different track and thus require an independent seek for the second object.

2.2 The solution: Packaging

Packaging combines contiguous placement with prefetching. Specifically, all the objects comprising a web page are stored contiguously on disk. The request for the web page comes with a size attribute that equal the size of the skeleton and all the embedded objects. As a result, the entire web page is read from disk in a single access.

Once a page has been read from disk, the server can parse it and cache all contents other than the skeleton, returning only the latter the (unaware) client. Subsequent requests for the embedded objects are granted from cache. Alternatively, the server may send all the content to the client, which must be aware of the technique. We next briefly address several complications:

Shared objects. An object may appear in multiple web pages. One solution is to replicate it, as storage capacity is inexpensive.

* Work done while at the Technion

In extreme cases, it may be left out of the package. If it is large, the extra cost of accessing it independently is insignificant. If it is small and very common, it can be cached.

Modifications. Whenever an object is modified, simply reading a package containing a stale version is erroneous. One solution is for the server to check whether it is current, just like a browser does when it has a cached version. If not, the server can modify the stored version or leave it, but in any case keeps a copy of the new version until all instances have been updated. Details are omitted for brevity.

We constructed a prototype using an Apache web server, and showed that packaging works.

2.3 Packaging: the Energy Perspective

The power consumed by a disk drive is the sum of three main components: 1) the motor that spins the disk, 2) the actuator doing the seek operation, and 3) the electronics for reading, writing, buffering and communicating the data. Packaging saves energy directly and indirectly, as explained below using two scenarios, each with and without packaging.

Equal workloads (same page-request traces). Consider a fixed time interval. Components 1 and 3 are equal (3 because the same data is read and transferred), but 2 is smaller with packaging because there are far fewer seek operations.

Equal disk utilization (fraction of time during which disk is busy). Assuming a busy server and typical pages, queuing delay is likely to dominate user-visible response time, which is a good measure of quality of service. Further assuming that the disk is the bottleneck, and since queuing delay is tightly linked to the load, i.e., to disk utilization, this is "equal-QoS" setting is an interesting one for a fair comparison. We again consider same-length time intervals, but now examine the amount of disk energy per request. 3 is again equal, and 2 is again smaller by the same factor. However, now 1 is also smaller, as the total energy in 1 is the same, but it is now amortized over a larger number of web pages that were read during the same time interval! The energy of some of the other server components per user request is also lower, but details are beyond the scope of this work.

3. Maximizing Delay-Constrained Throughput of Multi-Channel ALOHA

3.1 Background and Problem Statement

A multi-channel ALOHA network comprises a large number of upstream contention channels used by clients to send packets to a hub, and a private downstream channel for transmission of acknowledgements and data from the hub to clients. A client transmits a packet over a randomly chosen channel. The absence of an immediate Ack from the hub indicates a collision, and the client retries over a randomly chosen channel. ALOHA is used for short messages over broadcast infrastructure wherein round-trip time is much longer than packet transmission time, rendering channel sensing useless. Currently, an important application of the above is transaction processing via geostationary satellite.

The service provider pays a fixed fee for the satellite channels, and is rewarded per transaction. His goal is therefore maximum throughput with a given set of channels. The user's apparent goal is minimum delay, so the goals appear to be conflicting. Careful examination, however, reveals that the user actually has a maximum-delay (deadline) requirement. Considering a cash register example, the exact delay does not matter as long as the receipt is ready when the cashier wishes to get it, having first returned the credit card to the customer. So, the problem can be restated as delay-constrained throughput maximization. Unfortunately, ALOHA is a probabilistic scheme, so one must permit a certain probability of not meeting the deadline. This can be set 10 or 100 times smaller than the probability of other failures, such as dropping the card on the floor.

3.2 Solutions

Consider an example wherein the deadline permits a single retransmission attempt upon failure of the original attempt (a total of two attempts), and a permissible probability of failing to meet the deadline equal to 10^{-3}. For facility of exposition, assume that the probability of packet collision is 0.1. We next consider two schemes.

Minimum-delay scheme. The client randomly selects three different channels and transmits a copy of the packet over each of them. The probability of failure in this attempt is $0.1^3=10^{-3}$, as required. Moreover, delay is minimized, as all is done in the first attempt. However, three copies are always transmitted, taking up three channel time slots.

Maximum-throughput scheme. The client transmits a single copy in the first attempt, and two additional ones only upon failure. The deadline is still met with the required probability, but the mean number of copies per message drops to 1.2. This, in turn, means that the offered load (and throughput) can be increased by a factor of 2.5 with the same packet collision probability! (Optimization of the working point yields an even greater improvement.)

3.3 ALOHA: the Energy Perspective

The transmission energy per transaction equals the mean number of copies transmitted on its behalf. So, we managed to achieve a dramatic increase in throughput while commensurately **reducing** transmission energy. The key to this "win (throughput), win (energy), no-lose (delay)" was tailoring the scheme to the exact delay-related requirement.

We used the point-of-sale example, but battery operated sensors transmitting short, delay-critical packets to a central hub are another application, for which the energy savings may be critical.

References

[1] Y. Birk and Y. Keren, "Judicious Use of Redundant Transmissions in Multichannel ALOHA Networks with Deadlines," *IEEE J. Sel. Areas Commun. (JSAC)*, vol. 17(2), Feb. 1999.