Disk Delay Lines

Philip Machanick
Department of Computer Science
University of the Witwatersrand
2050 Wits, South Africa
philip@cs.wits.ac.za

The most important latency goals often relate to response times seen by users, which are slow by computer standards, but scaling up to large numbers of users presents a problem. Examples include transaction-based systems and web sites.

Here let us work through specific numbers; generalization is not too hard. The requirements are 100,000 transactions per second, maximum response time 1s. Further, assume that the disk may occupy a maximum of 0.5s of that response time. Assume that for 1Gbyte of data a disk with latency 7ms and transfer rate 40Mbyte/s is available, and a typical transaction reads 128 bytes (writes are a complication dealt with later), an amount small enough to suggest worrying about latency rather than fast streaming. The basic requirement is clearly out of reach. Although a response time of 0.5s (disk time only) is much larger than 7ms, the requirement of 100,000 transactions per second requires that each transaction take at most 10µs, almost 1000 times faster than the disk's access time. But what of the disk's 40Mbyte/s transfer rate? That looks more promising, as that allows 128 bytes to transfer in 3µs. The problem is that we cannot stream the data continuously since we require random accesses … but let's carry on with this line of thought.

What happens if we stream the disk continuously? Assuming this can be done with no pauses, the time to sweep the entire data set is 25.6s, still not so promising, since we want an operation to take at most 0.5s. What we have is 50 times too slow, in terms of user-level response time. Next step: replicate the data on 50 disks, synchronized so the data is timed to be equal distances apart as the disks stream, which should be possible, given that synchronizing disks is a solved problem for RAID. Now, any one item to be read is at most 0.5s away at any given time. The next trick is to queue requests (assuming the time for this is trivial compared with 0.5s) in a tagged buffer, which can detect when a given request matches the address of data being streamed off the disk (similar to cache tags). If the buffer can hold 50,000 requests, then up to 100,000 can be dispatched per second, assuming the worst case, that every request waits the maximum delay.

Writes present a more difficult problem. They could be buffered, and to avoid losing bandwidth or latency to them, ideally the drives should have a separate set of write heads. Obviously any reads that refer to buffered writes should pick up the buffered copy. Assuming the write problem can be handled, if a compute server can keep up with the required rate of transaction handling and the network interface is up to it (someone else's problem, only the disk subsystem is considered here), this design meets the stated requirements.

The competition is RAID [Chen *et al*. 1994], RAIS (redundant array of inexpensive servers), RAM-based databases and large-scale or mainframe-style disk systems. RAID cannot deliver the required latency. RAIS is used for large web sites; in principle, a RAIS system (or other distributed design) could better this design, but presents hard design problems like maintaining data consistency (in the presence of writes) and partitioning workload. A RAM-based solution gives better latency more easily and could beat the peak bandwidth of this solution, but doesn't match its potential fault tolerance or its nonvolatility across power failures. A high-end or mainframe-style disk subsystem is expensive and even so would have difficulty in achieving the design goals of the example presented here (under the assumptions here, 50 disks with 7ms access time couldn't support more than about 3,500 transactions per second if accessed the standard way).

The case made here is for disk designers for large-scale transaction-based systems to abandon the futile pursuit of lower latency and focus on maximizing bandwidth (e.g. by higher numbers of heads, and a faster interconnect to each disk). Large-scale web sites, in particular, with many fewer writes than reads, could benefit from this disk architecture.

Is the idea so new? Not really, a memory that continuously streams sequentially is one of the oldest ideas in digital computers: some early computers used an acoustic mercury delay line which had very similar properties [Bowden 1953].

Acknowledgments. This work was done while on sabbatical hosted by Trevor Mudge at Department of Electrical Engineering and Computer Science, University of Michigan. Financial support was provided by the University of the Witwatersrand and the South African Foundation for Research Development.

References.

[Chen *et al*. 1994] P.M. Chen, E.K. Lee, G.A. Gibson, R.H. Katz and D. A. Patterson. RAID: High-Performance, Reliable Secondary Storage, ACM Computing Surveys, vol. 26 no. 2 June 1994, pp 145-185.

[Bowden 1953] B. Bowden (ed.). Faster than thought, Pitman, London, 1953.