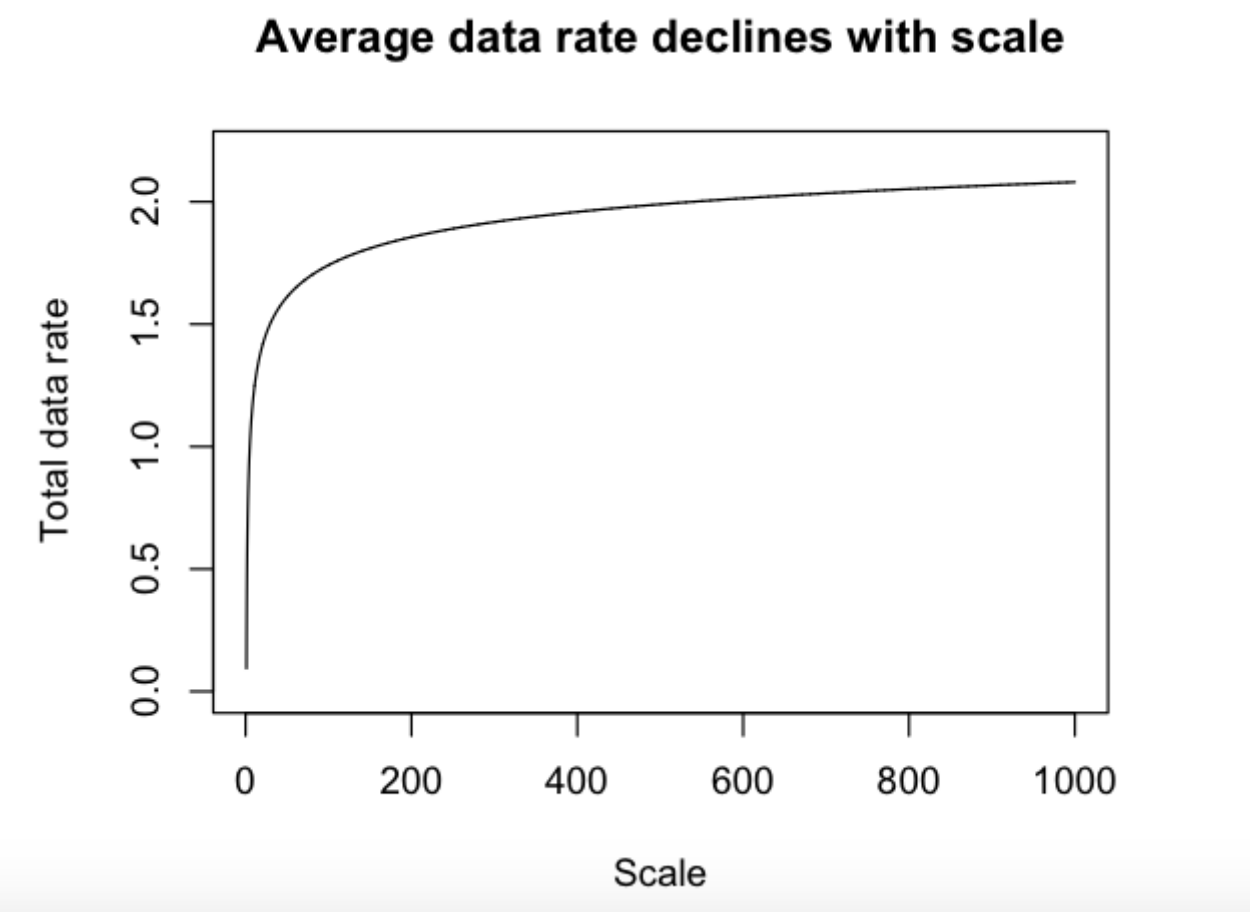
I commonly hear lots of questions about how many drives to use per node in a cluster. For a long time, the norm was to have 4-6 drives per node, but lately, I have been hearing more people suggest 12 drives. At MapR (https://mapr.com), we have been recommending 12 or 24 drives for quite some time to take advantage of the inherent advantages of MapR-FS (/blog/comparing-mapr-fs-and-hdfs-nfs-and-snapshots), but I still hear lots of people recommending smaller configurations.

In fact, I think that the norm is moving much higher than 12 drives. It is not uncommon for us to see boxes with up to 60 large drives lately. These are not the majority of systems by any stretch, but they have some very distinct advantages in terms of money (capex $/TB, opex $/TB) and power (opex W/TB).

For example, each drive consumes about 10W, while fully-populated motherboards consume about 350W. This means that a box with 6 drives puts 60W into spinning rust and 350W into CPU and stuff. That is obviously mostly CPU and costs 70W/drive. If you put in 60 drives, on the other hand, you are driving 600W into rust and 350W into CPU. That comes to 16W per drive or about 4x better power per TB.

The reason that this makes lots of sense is that bandwidth per byte follows the standard diminishing rate kind of curve. This means that as you expand total storage for any given application, you'll find that the amount of access per marginal byte of data drops dramatically. In any given application such as billing, there is a lot of activity when data is new, again on the month boundary and then again at year end, but the average rate dramatically drops as size increases.



The overall effect of this is to heavily cant the cost effectiveness balance toward large boxes for mature big data applications relative to early efforts. This goes well beyond the expectations that you would gain from early applications.

If you move this direction, you have to consider the software platform very carefully. For instance, HDFS- based systems can have very difficult times with large storage because of the cost of block reports and the difficulties in managing heterogeneous clusters. With MapR-FS, large nodes work much better because the basic architecture is designed to be much more scalable. Since containers manage most disk allocations completely locally, you avoid the problem of block reports. MapR-FS is also inherently able to drive far higher data rates, which is important when you have a large number of drives. We have recently seen systems that drive well in excess of 2GB/s second.

MapR-FS also allows you to control where data resides in a cluster. This allows you to build clusters with different kinds of machines. You might, for instance, have 20% of the nodes in a cluster have very deep storage with the remainder being used much more for compute intensive tasks. Obviously, in such a cluster you would like to make sure that archival, low access-rate data is what goes onto the deep storage nodes and hot data is spread around more evenly. MapR-FS provides this capability natively by allowing volumes to be restricted to particular nodes.

Overall, it is very important to consider the entire design space when you start to build a production cluster. You should consider whether your workload characteristics will change beyond prototyping scale and whether you might be able to take advantage of substantial savings by using large nodes.

This blog post was published December 18, 2014.