Model of the microcode implementation of the Fully Automated Storage Tier (FAST) adaptive algorithm for simulation purposes

D. Gueorguiev

10/5/15

Overview:

Described is the model and algorithm employed by the simulation tool *WorkloadPlanner* in order to simulate the microcode implementation of the *Fully Automated Storage Array* adaptive algorithm which decides how an incoming workload with given skew will be distributed on the target pools. Also in the process of the calculation it becomes clear if incoming workload is admissible or not. The algorithm attempts to achieve optimal load placement on the target pools such that the total SLO/SLE measure (see ***Appendix A1***) will reach local minimum.

Notation:

LS^{inc} – the incoming load score with given skew. It is given with a set of skew chunks represented as load score-capacity pairs $\{LS_1^{inc}, C_1^{inc}\}, \{LS_2^{inc}, C_2^{inc}\}, \dots, \{LS_K^{inc}, C_K^{inc}\}$

$LS_{SG,P}$ - the load score of storage group SG on pool P . The load score is defined as $4 \times \left(read\ misses + \frac{prefetches}{2}\right) + 5 \times \left(write\ misses + \frac{prefetches}{2}\right) + 11 \times MBs\ transferred\ per\ \text{sec}\ on\ the\ back\ end$

Obtained from `SGByPool` or `SGByTier` metric category

$C_{SG,P}$ - the capacity in GB of storage group SG on pool P. Obtained from `SGByPool` or `SGByTier` metric category

$\overline{AD}_P^{tot}$ - total average access density per pool (defined in step 1)

LN_P – the lossiness for pool P (defined in step 3).

$PRT_P^{(n)}$ – the pool response time for pool P on iteration n (defined in step 4).

$PRT_P^{(SLE)}$ – the SLE response time for pool P

$SRT_{SG}^{(n)}$ – the storage group response time for storage group SG on iteration n (defined in step 5).

$SRT_{SG}^{(SLO)}$ – the SLO response time specified for storage group SG

$PRF_{SG}^{(n)}$ – the priority factor for storage group SG on iteration n (defined in step 6).

$FLS_{SG,P}^{(n)}$ – fast (effective) load score for storage group SG on pool P for iteration n (defined in step 7).

$FLF_{SG}^{(n)}$ – fast (effective) load factor for storage group SG on pool P for iteration n (defined in step 8).

N_{iter}^{max} – max number of iteration before the algorithm terminates if it did not converge already.

The Algorithm:

1. Calculate the total average access density per pool:

$$\overline{AD}_P^{tot} = \frac{\sum_{SG} LS_{SG,P}}{\sum_{SG} C_{SG,P}}$$

2. Calculate the initial priority factor per storage group:

$PRF_{SG}^{(0)} = \sum_P \frac{AD_{SG,P}}{\overline{AD}_P^{tot}}$ where the access density of storage group SG on pool P is given with $AD_{SG,P} = \frac{LS_{SG,P}}{C_{SG,P}}$. Here $LS_{SG,P}$ is the load score of storage group SG on pool P and $C_{SG,P}$ is the capacity in GB of storage group SG on pool P.

3. Calculate the lossiness per pool:

$LN_P = \frac{PLS_P}{\sum_{SG} LS_{SG,P}}$ where PLS_P is the thin pool load score of pool P obtained from the `TPThinPool` metric category. Note that the thin pool load score is defined as $4 \times thin\ pool\ read\ misses + 5 \times thin\ pool\ write\ misses + 11 \times thin\ pool\ back\ end\ MBs\ transferred\ per\ sec$

4. Calculate pool response time on the n-th iteration ($n = 1..N_{iter}^{max}$)

$PRT_P^{(n)} = A + B \times \left(\frac{\sum_{SG} LS_{SG,P}^{(n-1)}}{PLS_P^{(SLE)}} \right)^{2.2}$

where $LS_{SG,P}^{(n-1)}$ denotes the SG-by-pool load score from the previous iteration. Obviously $LS_{SG,P}^{(0)} = LS_{SG,P}$ which is the initial SG-by-pool load scores obtained from the SPA `SGByPool` category.
Here $PLS_P^{(SLE)}$ is the pool load score for pool P at SLE response time. The values of $PLS_P^{(SLE)}$ for various disk technologies are calculated by using appropriately tabulated disk IO rates versus response time data a.k.a. *the disk model*.
A and B are coefficients determined by the conditions:

$PRT_P^{(0)} = A + B \times \frac{PLS_p^{(0)}}{PLS_P^{(SLE)}}$.

$PRT_P^{(SLE)} = A + B$

In the first relation above the quantity $PLS_p^{(0)}$ denotes the initial pool load score and is given as $PLS_p^{(0)} = \sum_{SG} LS_{SG,P}^{(0)}$. Also $PRT_P^{(0)}$ denotes the initial pool response time which is given and is pre-calculated using appropriate disk model assuming equal load distributions among the disks which belong to the pool P.

5. Calculate storage group response time on the n-th iteration ($n = 1..N_{iter}^{max}$):

$SRT_{SG}^{(n)} = \frac{\sum_P RM_{SG,P} \times PRT_P^{(n-1)}}{\sum_P RM_{SG,P}}$ where $PRT_P^{(n)}$ is the pool response time on the n-th iteration.
Here $RM_{SG,P}$ is the read miss per storage group per pool from the metric category `SGByTier` (V2) and `SGByPool` (V3) which follows Amnon's suggestion. Alternatively, instead of $RM_{SG,P}$ we can use $LS_{SG,P}$.

6. Calculate the scalar load multiplier $lm^{(n)}$ on the n-th iteration

First, calculate the scalar load multiplier lm by using the equality:

$PRT_P = A + B \times \left(\frac{\sum_{SG} LS_{SG,P}}{PLS_P^{(SLE)}} \times lm \right)^{2.2} = PRT_P^{(SLE)}$

where the quantities A, B, $PLS_P^{(SLE)}$ and $PRT_P^{(SLE)}$ were defined in 4.

$lm = \min_P \left\{ \frac{PLS_P^{SLE}}{\sum_{SG} LS_{SG,P}} \left(\frac{PRT_P^{(SLE)} - A_P}{B_P} \right)^{\frac{1}{2.2}} \right\}$

Then with the obtained value lm recalculate the adjusted response times for all pools

$$PRT_P^* = A + B \times \left(\frac{lm \times \sum_{SG} LS_{SG,P}^{(n-1)}}{PLS_P^{(SLE)}} \right)^{2.2}$$

With the adjusted pool response times PRT_P^* calculate the adjusted storage group response times:

$$SRT_{SG}^* = \frac{\sum_P LS_{SG,P}^{(n-1)} \times PRT_P^*}{\sum_P LS_{SG,P}^{(n-1)}}$$

Check if any of the adjusted storage group response time exceeds its SLO:

$$SRT_{SG}^* \leq SRT_{SG}^{(SLO)}$$

In case there is a storage group for which the inequality above does not hold then
then calculate the scaling factor δ which scales down the adjusted storage group response times and adjusted pool response times such that all SG SLO constraints are met:

$$SRT_{SG}^{**} = \delta \times SRT_{SG}^* \leq SRT_{SG}^{(SLO)}$$

Then the scaled down adjusted pool response time PRT_P^{**} is given with

$$PRT_P^{**} = \delta \times PRT_P^*$$

Using the scaled down adjusted pool response time PRT_P^{**} calculate the scaled down load multiplier which $lm^{(n)}$:

$$lm^{(n)} = \min_P \left\{ \frac{PLS_P^{SLE}}{\sum_{SG} LS_{SG,P}} \left(\frac{PRT_P^{**} - A_P}{B_P} \right)^{\frac{1}{2.2}} \right\}$$ which becomes the scalar load multiplier on the n-th iteration

7. Calculate storage group priority factor on the n-th iteration[1]:

$$PRF_{SG}^{(n)} = \begin{cases} PRF_{SG}^{(0)} \times \frac{SRT_{SG}^{(n)}}{SRT_{SG}^{SLO}} \text{ if } PRF_{SG}^{(0)} \times \frac{SRT_{SG}^{(n)}}{SRT_{SG}^{SLO}} > 1.0 \\ 1.0 \text{ if } PRF_{SG}^{(0)} \times \frac{SRT_{SG}^{(n)}}{SRT_{SG}^{SLO}} \leq 1.0 \end{cases}$$

8. Calculate FAST load score per storage group per pool on the n-th iteration:

$$FLS_{SG,P}^{(n)} = LS_{SG,P} \times \frac{PRF_{SG}^{(n)} + PRF_{SG}^{(n-1)}}{\sum_{SG} PRF_{SG}^{(n)} + PRF_{SG}^{(n-1)}}.$$

In the last equation the terms $PRF_{SG}^{(n)} + PRF_{SG}^{(n-1)}$ have dampening effect against oscillations in the resulting fast load score.

9. Calculate FAST Load Factor per storage group on the n-th iteration

$$FLF_{SG}^{(n)} = \frac{PRF_{SG}^{(n)} + PRF_{SG}^{(n-1)}}{\sum_{SG} PRF_{SG}^{(n)} + PRF_{SG}^{(n-1)}} \; ; \; FLF_{SG}^{(0)} = 1$$

10. Calculate Incoming load distribution on the pools LS_P^{inc}

The end result of this step is to obtain distribution of the incoming workload LS^{inc}
on the target pools $LS_P^{inc}, P = 1,2, ..$

[1] J.L. tried also $PRF_{SG}^{(n)} = PRF_{SG}^{(0)} \times \sqrt{\frac{SRT_{SG}^{(n)}}{SRT_{SG}^{SLO}}}$

Detailed discussion on the pool load distribution algorithm is beyond the scope of this document and only a brief explanation of how the pool load distribution works will be supplied here.

For each pool it is assembled a skew profile which is represented as an ordered by decreasing access density sequence of `SGByPool` skew chunks for all target storage groups.
The incoming workload is represented by its skew given as a set of pairs $\{LS_k^{inc}, C_k^{inc}\}$ with corresponding access density $AD_k^{inc} = \frac{LS_k^{inc}}{C_k^{inc}}$ which are added to the sequence of the sorted skew chunks. For the purpose of the pool load distribution process the access density of each skew point is multiplied by the fast load factor of the storage group it originates from as shown:

$$AD_k^* = AD_k \times FLF_{SG}^{(n)}$$

where AD_k^* represents the *adjusted* access density of the skew point which originates from the storage group SG.
All of the skew points from all pools are sorted by decreasing adjusted access densities. Let us denote the sorted sequence of skew points by $\langle\{AD_1^*, C_1\}, \{AD_2^*, C_2\}, \dots, \{AD_k^*, C_k\}, \dots, \{AD_K^*, C_K\}\rangle$.

Let us assume also that we have P target pools sorted by technology and SLE response times. Each pool is represented by the pair of pool response time at SLE point and the available free capacity $\{PRT_p^{SLE}, C_p\}$. Then the sorted sequence of pools is denoted by

$\langle\{PRT_1^{SLE}, C_1\}, \{PRT_2^{SLE}, C_2\}, \dots, \{PRT_p^{SLE}, C_p\}, \dots, \{PRT_P^{SLE}, C_P\}\rangle$ where $PRT_1^{SLE} \leq PRT_2^{SLE} \leq \cdots \leq PRT_p^{SLE} \leq \cdots \leq PRT_P^{SLE}$.

Also we will assume that the first P_{EFD} pools of the sorted pool sequence belong to technology EFD , the second tier of P_{FIBRE} pools belong to technology $Fibre$ and the third tier of P_{SATA} pools belong to technology $SATA$. Obviously, $P_{EFD} + P_{FIBRE} + P_{SATA} = P_{tot}$.

The next step in the pool load distribution process is fitting the sorted by adjusted access densities list of skew chunks into the available free capacity of the sorted list of pools. If the total capacity on the pools is smaller than the total capacity of the skew chunks then the load distribution algorithm generates ***ERROR_INSUFFICIENT_POOL_AVAILABLE_CAPACITY*** and the back end admissibility service immediately returns negative 1. However if all of the skew chunks fit into the target pools on the same SRP with respect to capacity then it is possible that the incoming workload is admissible and the processing moves to the next step of the pool distribution process. Let us denote by S_p the set of chunks which land on pool P_p, where $p = 1..P$. Note that some chunks may have been split between the pools or if large enough even can land on several different pools. A list of leftover chunks is maintained which contains all chunks which did not land on a pool during the first pass of the sorted chunks distribution either because of SRP mismatch or because of violating SLO constraint. A second pass over the target pools is performed with the chunks in the leftover list attempting to distribute those by displacing an already distributed chunk which SLO allows it to be demoted to a lower tech pool or promoted to a higher tech pool. Any evicted chunk which cannot be found a place satisfying its SLO and SRP constraints is placed in the list of the leftover chunks. A third pass over the pools is performed with the updated list of leftover chunks looking for pools with free space which can accommodate the leftover chunks without violating their SLO and SRP constraints. If the third pass fails to distribute all leftover chunks the load distribution algorithm generates ***ERROR_INSUFFICIENT_POOL_AVAILABLE_CAPACITY*** and the back end admissibility service immediately returns negative 1. However, if all of the leftover chunks have found place on some pool

such that their SLO and SRP constraints are not violated then it is possible that the incoming workload is admissible and the resulting incoming load distribution LS_P^{inc} is obtained.
Let us denote the incoming load distribution on the pools for the incoming chunk k with $LS_{k,P}^{inc}$. Note that $LS_k^{inc} = \sum_P LS_{k,P}^{inc}$ is the original load score of the incoming chunk k and it is not the adjusted load score $LS_k^{inc\,*} = LS_k^{inc} \times FLF_{SG}^{(n)}$ which was used only to determine the current incoming load distribution. Then the total incoming load pool distribution will be given with $LS_P^{inc} = \sum_k LS_{k,P}^{inc}$. After the distribution is done insert the incoming chunks on the target pools and take them into account when calculating the pool response times $PRT_P^{(n+1)}$ and the storage group response times $SRT_{SG}^{(n+1)}$ on the next iteration.

Use the FAST load score/factor to recalculate the pool skew and re-run the pool load distribution thereby effectively modifying the pool load score and pool response times to new sets of values $PLS_p^{(n+1)}$ and $PRT_P^{(n+1)}$. Repeat from step 4. onward until $n \le N_{iter}^{max}$ or desired convergence for the storage group response times $SRT_{SG}^{(n)}$ and the pool response times $PRT_P^{(n)}$.
If $SRT_{SG}^{(n)} \le SRT_{SG}^{(SLO)}$ for every target SG and If $PRT_P^{(n)} \le PRT_P^{(SLE)}$ for every pool then incoming workload is admissible in the system. In case storage group response time do not converge as the iteration increase then the final incoming load distribution is the one from the iteration which has minimal value of the total SLO/SLE measure (described in the Appendix).

As soon as the calculation completes and a final load multiplier lm is found calculated the bucketized load multiplier using the expression:

$\overrightarrow{LM} = \frac{lm \cdot \sum_p \overrightarrow{LS}_p - \sum_p (\overrightarrow{LS}_p - \overrightarrow{LS}_p)}{\sum_p \overrightarrow{LS}_p}$ which is simplified to

$$\overrightarrow{LM} = lm - \left(\frac{\sum_p \overrightarrow{LS}_p}{\sum_p \overrightarrow{LS}_p} - 1\right)$$

Then the pool admissibility buckets $\overrightarrow{PA}$ is calculated as:

$$\overrightarrow{PA} = 1 - \frac{1}{\overrightarrow{LM}}$$

Appendixes

Appendix A1

The total SLO/SLE measure M_{tot} is defined as $M_{tot} = \sum_{SG} \frac{SRT_{SG}}{SRT_{SG}^{SLO}} \times 100 + \sum_P \frac{PRT_P}{PRT_P^{SLE}} \times 100$
It measures how close in percents the SGs are to the respective SLOs and the pools are to their respective SLEs. Note that in order the incoming workload to be admissibile the total SLO/SLE measure $\le (N^{SG} + N^P) \times 100$ where N^{SG} is the total number of storage groups and incoming workload chunks and N^P is the total number of pools. If the pool capacity is a concern then a third term reflecting the used pool capacity and the price per GB per pool can be added to the total measure as shown below:

$M_{tot} = \sum_{SG} \frac{SRT_{SG}}{SRT_{SG}^{SLO}} \times 100 + \sum_{P} \frac{PRT_P}{PRT_P^{SLE}} \times 100 + \sum_{P} \frac{C_P^{used}}{C_P^{tot}} \times \alpha_P$ where C_P^{tot} denotes the total disk capacity available at pool P , C_P^{used} is the total used capacity of pool P and α_P is the price per GB for the pool P.

Appendix A2

Appendix A3

FAST load score variant used by J.L. simulation engine:

$FLS_{SG,P}^{(n)} = (LS_{SG,P} - RM_{SG,P}) \times \overline{SF}^{tot} + RM_{SG,P} \times \left(PF_{SG}^{(n)}\right)^3$

where the total weighted SG priority factor $\overline{SF}^{tot}$ is given with

$\overline{SF}^{tot} = \frac{\sum_{SG} PF_{SG}^{(n)} \times RM_{SG}}{\sum_{SG} RM_{SG}}$

Another variant for FAST load score used by J.L.:

FAST Load score variant used by J.L. simulation engine:

$FLS_{SG,P}^{(n)} = PF_P \times LS_{SG,P} + PRF_P^{(n)} \times RM_{SG,P} \times 4$

where the pool factor PF_P is given with

$PF_P = \frac{IO_P}{IO_P^{SLE}}$. Here IO_P is the total IO rate per disk for pool P and IO_P^{SLE} is the max pool IO rate from the pool SLE.