A New Metric for Analyzing Storage System Performance Under Varied Workloads

Touch Rate

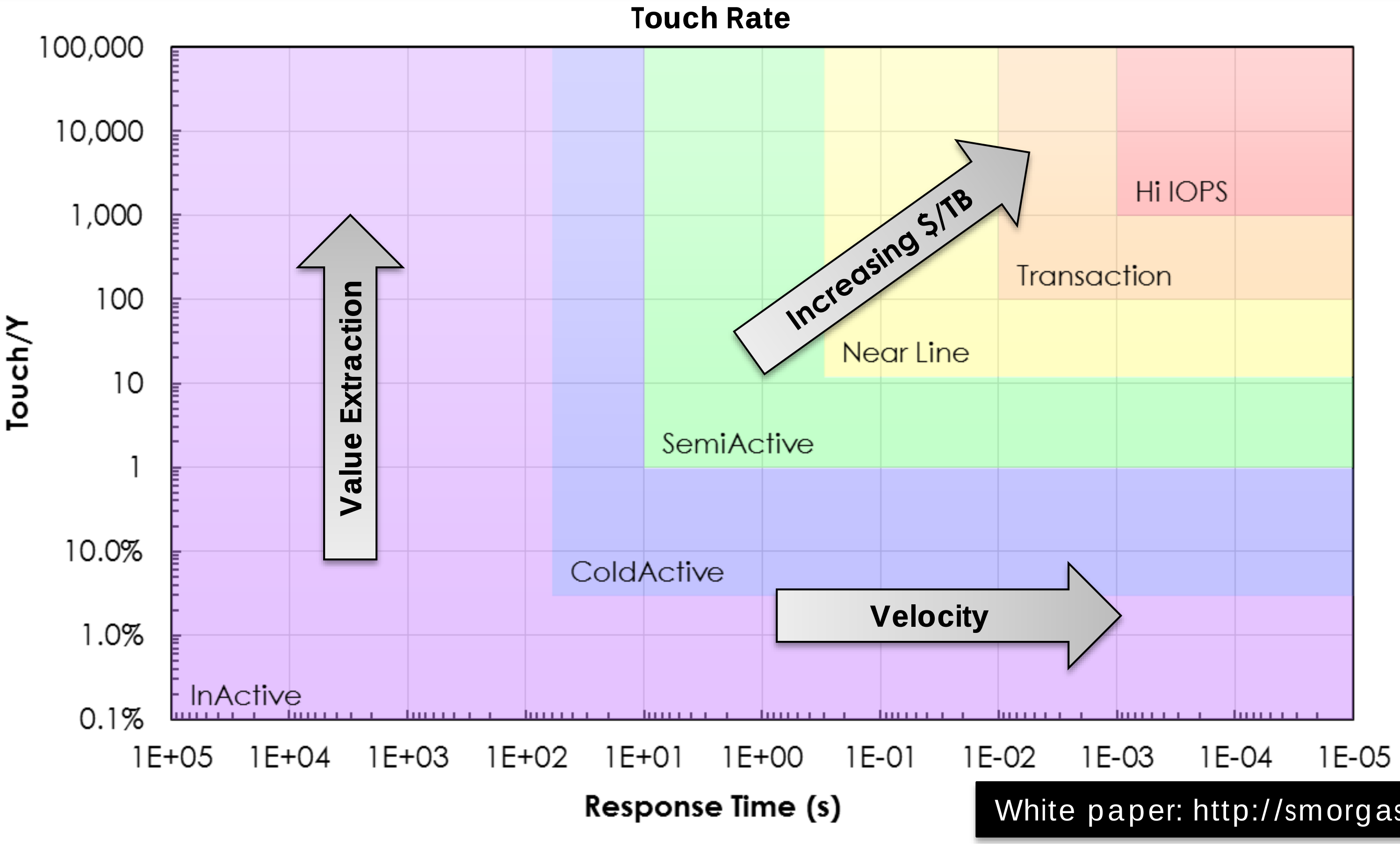
Steven Hetzler
IBM Fellow
Manager, Cloud Data Architecture

Overview

- New method for analyzing storage system behavior – Touch Rate
- This can be used to quantify the performance characteristics of flash storage systems
- We will look at the following classes of systems:
  - Disk
  - Flash
  - Hybrid systems
- All performance values shown are for active systems
  - Assume back-to-back IOs (no idle time)
    - Cloud systems are never idle (systems with idle time are overprovisioned)
  - Queuing is not addressed here
- Analysis is for read operations on flash
  - Write operations need to account for drive writes per day limits

Touch Rate Regions

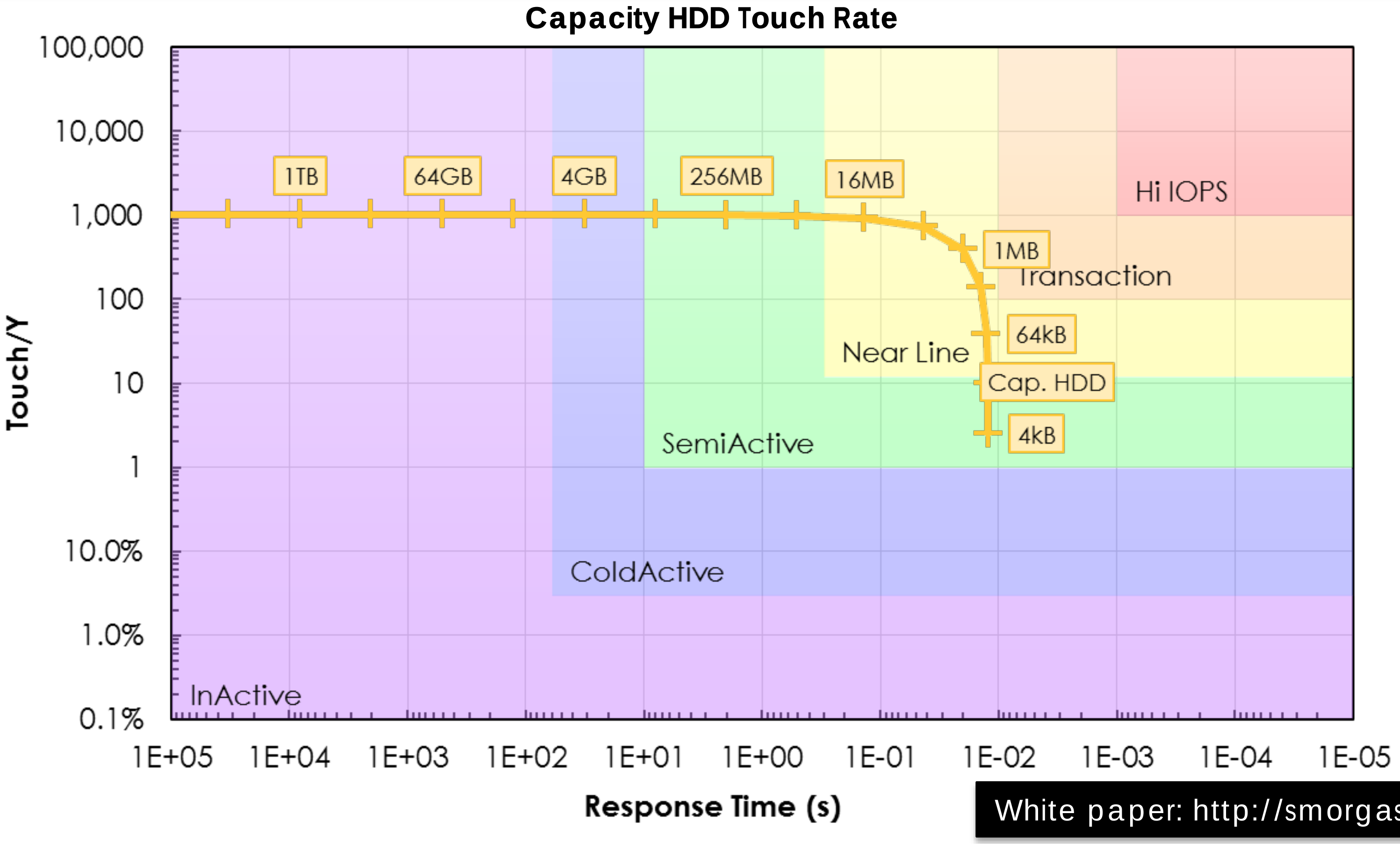
- Touch rate is fraction of total data (device or system) accessible per unit time
  - It's inventory turns on the data set – **value that can be extracted from data**
  - Scale independent
- Response time is time to complete IO in busy system (back-to-back IOs) – **velocity**
- (See the white paper for equations)



White paper: http://smorgast[illegible]

Touch Rate Curve

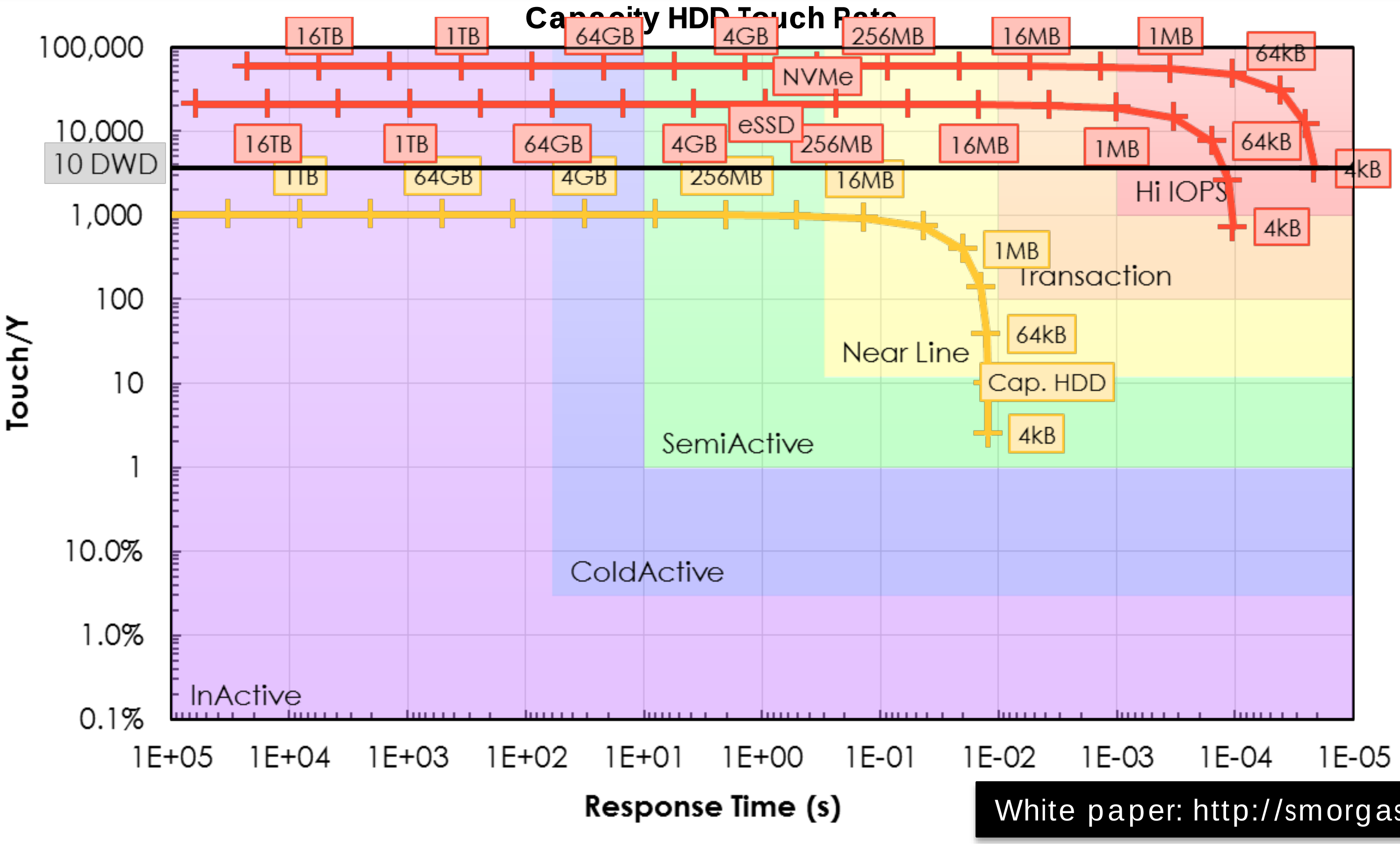
- Touch curve shows effect of IO object size
- IO object size is a characteristic of the workload
- Here, system of 4TB capacity optimized HDDs



White paper: http://smorgast

Flash Device Touch Rate

- Flash storage exhibits substantial improvements in touch rate over HDD
- eSSD is a 1.6TB 12Gb SAS SSD
- NVMe is a 1.6TB PCI-e card with NVMe interface
- Flash better where value extracted from data exceeds that from HDD



White paper: http://smorgast

Hybrid System Design

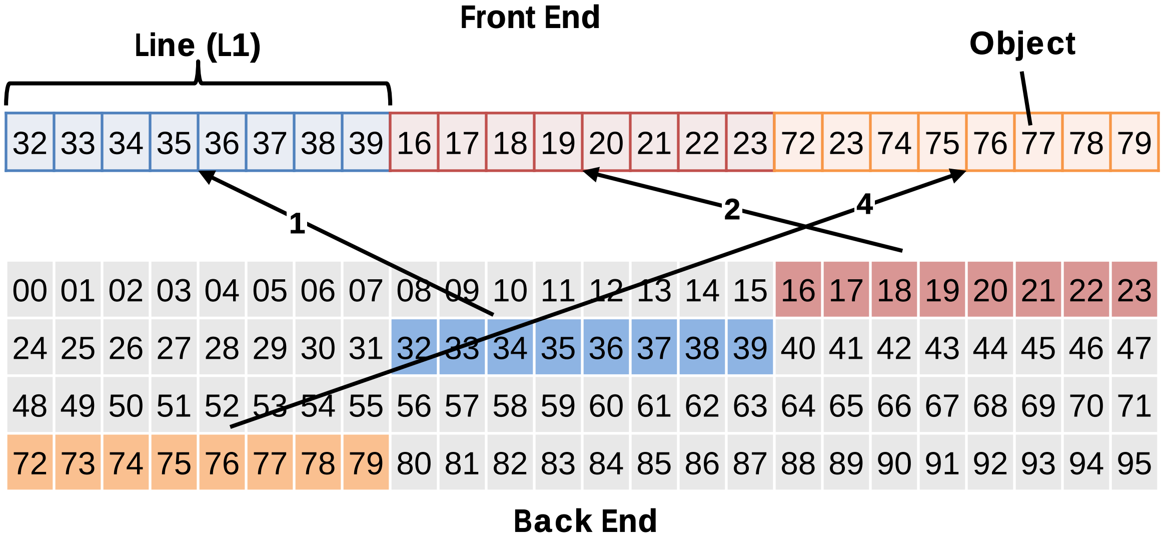
- Let’s examine flash/HDD hybrid system design
  - Performance depends on both layers
  - This is a simple example to explain the method
- Assume we use capacity HDD for back end storage
  - 15 x 4TB in RAID 6, so 13TB user capacity
  - 10GbE attached (1GB/s bandwidth)
- Flash for front end
  - Look at SAS SSD and NVMe
- We need to understand performance for a mixed technology system
  - Analysis is similar for caching and tiering
  - Compare performance of flash interfaces
  - Hit ratio is % of IOs serviced from data resident in the front end

Caching/Tiering

- Using a small amount of a faster technology (the front end - FE) and a slower large backing store (the back end - BE)
- Value and cost of caching/tiering depends on the characteristics of front end, back end and the workload
  - – Caching helps, but BE performance limits the practical gains

Example:

1. Get(34) : Load line(32) to front end L1, return(34)
2. Get(22) : Load segment(16) to front end L2, return (32)
3. Get(36) : read (36) from front end, return(36) ← cache hit
4. Get(78) : Evict L3, load segment(72) to front end L3, return(78)
5. …



Front End

32	33	34	35	36	37	38	39	16	17	18	19	20	21	22	23	72	23	74	75	76	77	78	79

Back End

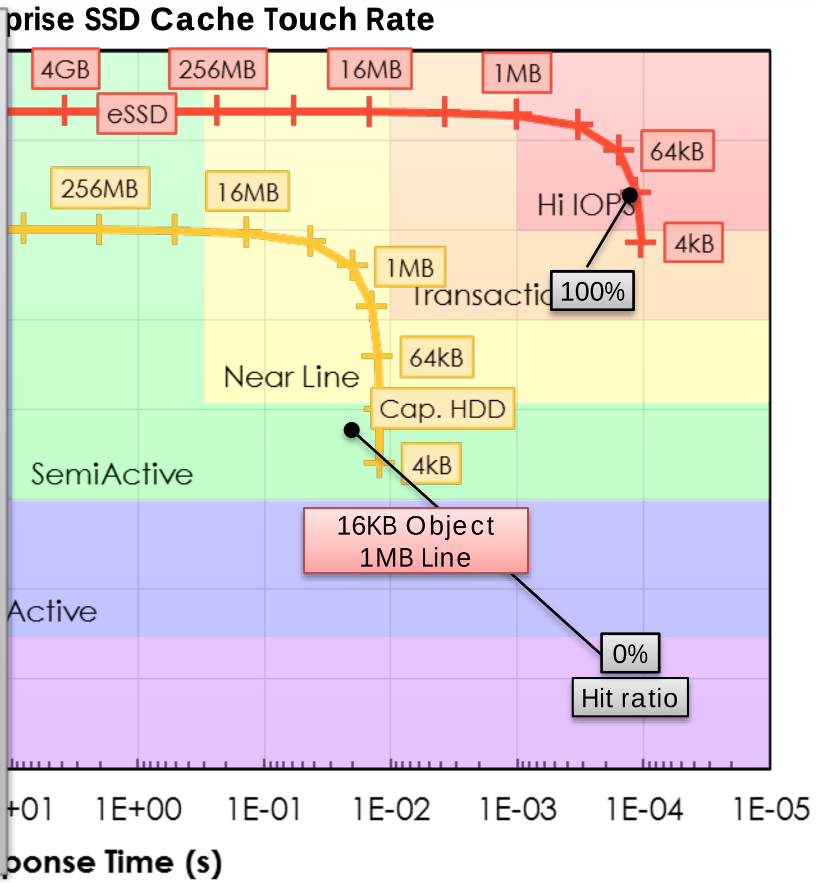
00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95

Caching/Tiering Touch Rate

- Caching and tiering are very similar
  - Cache line = tier segment
- Computing touch rate for back end and a front end
  - 2 object sizes of interest:
    - The application object size (font end IOs are this size)
    - The line size (back end IOs are this size)
  - Hits have front end response time @ object size
  - Misses have back end response time @ line size
    - A new line must be loaded to the front end
    - And another line must be evicted from the front end to make room
  - For a busy system , we assume a miss always loads a line
    - Not always true for caches
  - Consider only read caching here
- Plots are net touch rate

Touch Rate Caching Curve

- Can compute the rate curve for a given object size and line size
  - Here, front end is eSSD, back end is capacity HDD
  - Assume an object size of 16kB and a line size of 1MB.
- Compute response time vs. hit ratio for 16KB objects
- We expect the 100% hit ratio point to be at the object point on the front end curve
- The 0% hit ratio point would be below and left of the back end curve at the line size response time

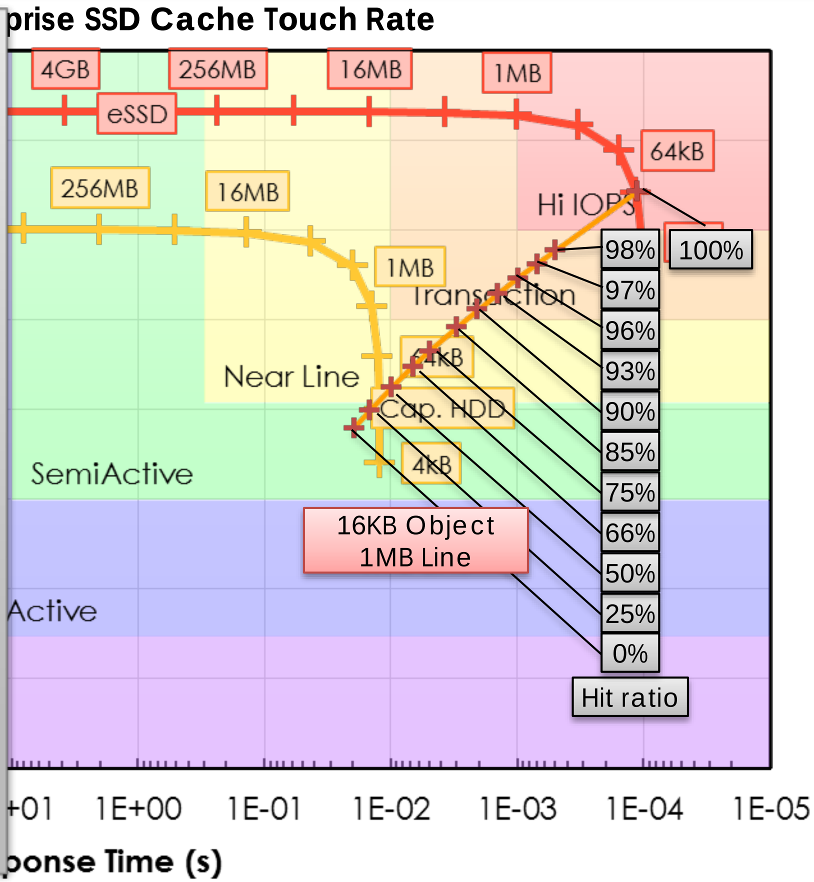


- In this model, the system is assumed busy, thus the penalty for loading a line on a cache miss is charged to the IO
  - Tiering loads the line on migrate
- This is a simple example for read only
- More complex designs can have different behavior

Touch Rate Caching Curve

- Can compute the rate curve as a function of hit ratio
- Plotted as a contour, with hit ratio points plotted
  - The points are not linearly distributed
- We see a performance gain here if the hit ratio is >~50%
- Achieving a hit ratio depends on the workload and the ratio of the front end to back end capacities
  - Thus, higher hit ratios are more expensive

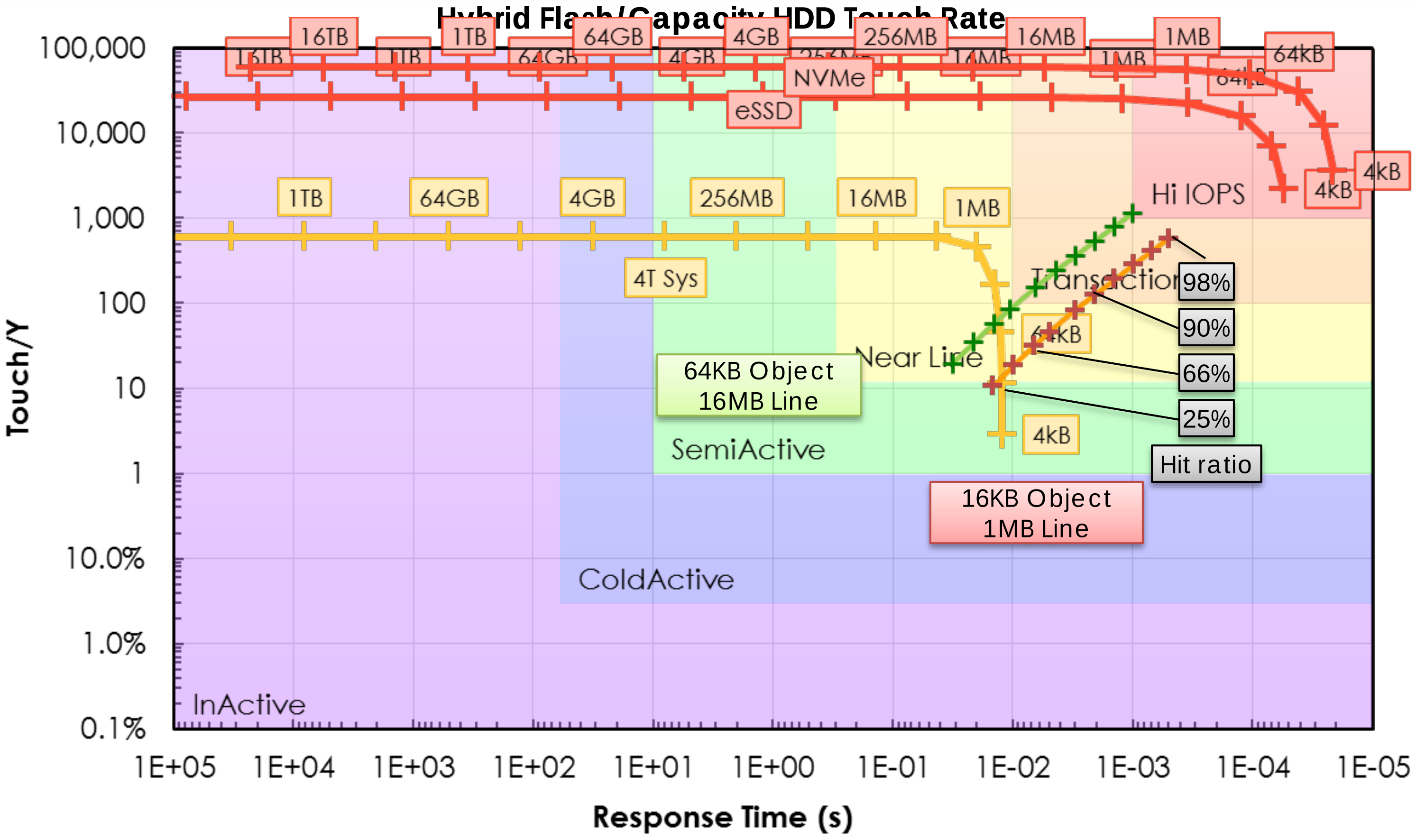
prise SSD Cache Touch Rate



- Contour parameter is cache hit ratio
- Hit ratio points are roughly 1/x distributed
- Thus the gains diminish with hit ratio
- Not all designs will be economical

Hybrid Flash Touch Rate

- Capacity HDD system is touch rate limited due to network BW
- Orange curve is for 16kB object and 1MB line size
  - At hit ratio > 50%, performance is improved
- Green curve is for 64kB object and 16MB line size
  - Need hit ratio >66% to improve performance



Analysis

- Flash can improve the touch rate for hybrid systems
  - Results shown hold for either tiering or caching
  - I have modeled basic caches – more advanced designs can do better
- Back end performance dominates
  - Increasing the front end performance helps only at high hit ratios
  - The further the front end and back end separation, the harder it is to see benefits from faster front ends
    - FLAPE is prime example – caching tape to flash is no faster than to HDD until well past a 99% hit ratio
- Choice of SSD vs. NVMe front end is not clear
  - High hit ratios clearly favor NVMe (>90%)
  - You need to look at the solution costs to see which is best for you
- The analysis here can be applied to other hybrid systems

Conclusion

- I have introduced Touch Rate for cached or tiered systems
- Tom Coughlin and I will cover this in detail in our next white paper
- I will release an updated spreadsheet including the ability to compute cached/tiered touch rate when the new white paper is released
- You can get a copy of the current white paper and spreadsheet at:
  - http://smorgastor.drhetzler.com/library
  - A copy of this talk will be also posted there
- FMS 2015 bonus:
  - Download the advanced version 1.4 of the spreadsheet
    - Can automatically create touch rate plots at
  - http://smorgsator.drhetzler.com/library/fms2015
    - (The macros are released as open source <and might even work>)