BIG DATA THRASHING

RESOLUTION THROUGH

MACHINE LEARNING

Presented by: Rasananda Behera

BIG DATA ARCHITECT/ CHIEF ENTERPRISE ARCHITECT

JUNE 6-8 2017 JAVITS CENTER, NY @ CLOUDEXPO

THRASHING - WHY

A particularly troublesome phenomenon, thrashing, may seriously interfere with the performance of paged memory systems, reducing computing giants to computing dwarfs. The term thrashing denotes excessive overhead and severe performance degradation or collapse caused by too much paging. Thrashing inevitably turns a shortage of memory space into a surplus of processor time.

Thrashing occurs:

- When a computer's virtual memory subsystem is in a constant state of paging,
- Rapidly exchanging data in memory for data on disk, to the exclusion of most application-level processing.
- Thrashing is caused by under allocation of the minimum number of pages required by a process, forcing it to continuously page fault.
- The system can detect thrashing by evaluating the level of CPU utilization as compared to the level of multiprogramming. It can be eliminated by reducing the level of multiprogramming.

Resolve thrashing due to excessive paging, a user can do any of the following:

[1] Increase the amount of RAM in the computer.

[2] Decrease the number of programs being run on the computer.

[3] Replace programs that are memory-heavy with equivalents that use less memory.

SWAPPING

Swapping concept comes in terms of process scheduling.

- Swapping is basically implemented by Medium term scheduler.
- Medium term scheduler removes process from CPU for duration and reduce the degree of multi-programming segment of a program in memory (RAM) with another part of the program and restoring it back to the original if required.
- Swapping can be implemented in various ways. For example, swapping can be priority based.
- If a higher priority process arrives and wants service, the memory manager can swap out the lower priority process to the secondary memory
- Higher priority process can be loaded to main memory for execution. Higher priority process finishes, the lower priority process will be swapped back to main memory and execution will be continued. Sometimes swapping is also called roll out, roll in.
- Virtual Memory swapping can have a large impact on the performance of a Hadoop system.

SWAPING PAGE PROBABILITY

[GOOD OR BAD]

A good measure of performance for a paging policy is the missing-page probability, which is the probability that, when a process references its program, it directs its reference to a page not in main memory.

The better the paging policy, the less often it removes a useful page, and the lower is the missing-page probability. We shall use this idea to examine three important paging policies (ordered here according to increasing cost of implementation) :

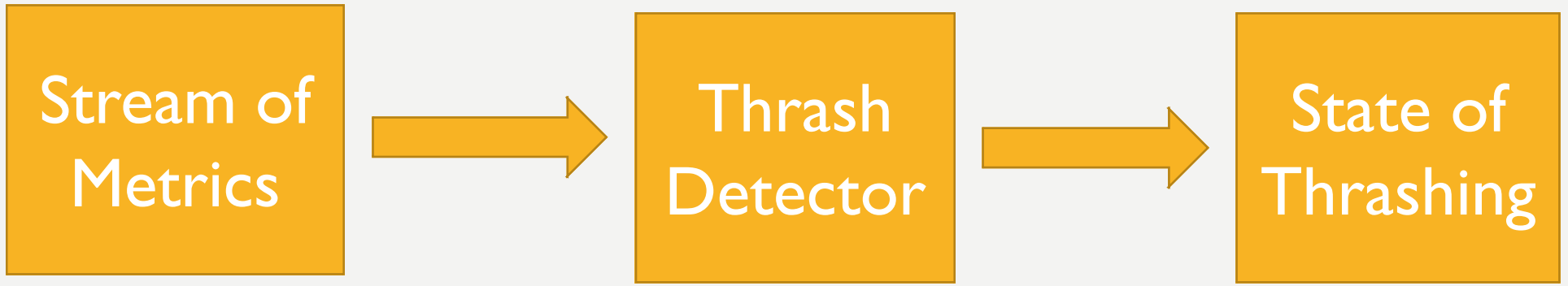
1. First In, First Out (FIFO): Whenever a fresh page of main memory is needed, the page least recently paged in is removed.

2 Least Recently Used (LRU): Whenever a fresh page of main memory is needed, the page unreferenced for the longest time is removed.

3. Working Set (WS): Whenever a fresh page of main memory is needed, choose for removal some page of a non-active process or some non-working set page of an active process.

Let's detect and examine…

DETECTION I/P & O/P



- [0] Zero Chance
- [1] High Chance
- [2[Active

REMEDY

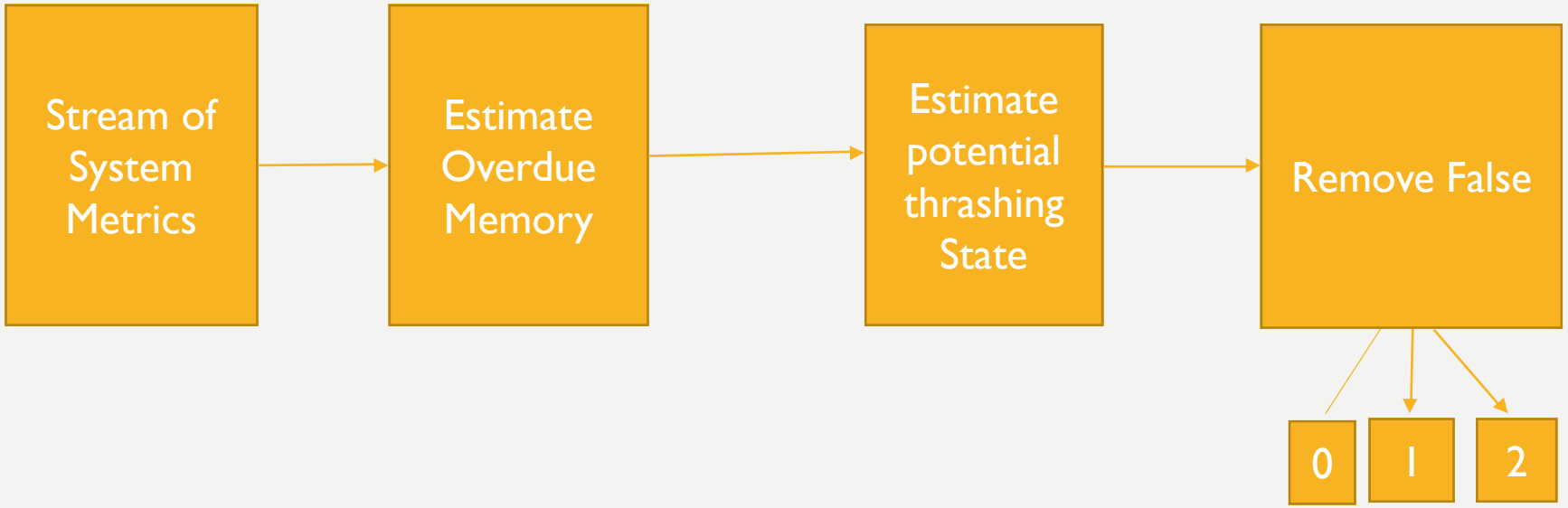
[I] Observe Per System with Granularity

a) Swap In & Swap Out
b) Free Memory

[II] Supervised Learning Process (Detect)

[III] Automatic Process (Machine/System act)

REMEDY CONTD…



LEARN PARAMETERS: MINIMIZE SCORE

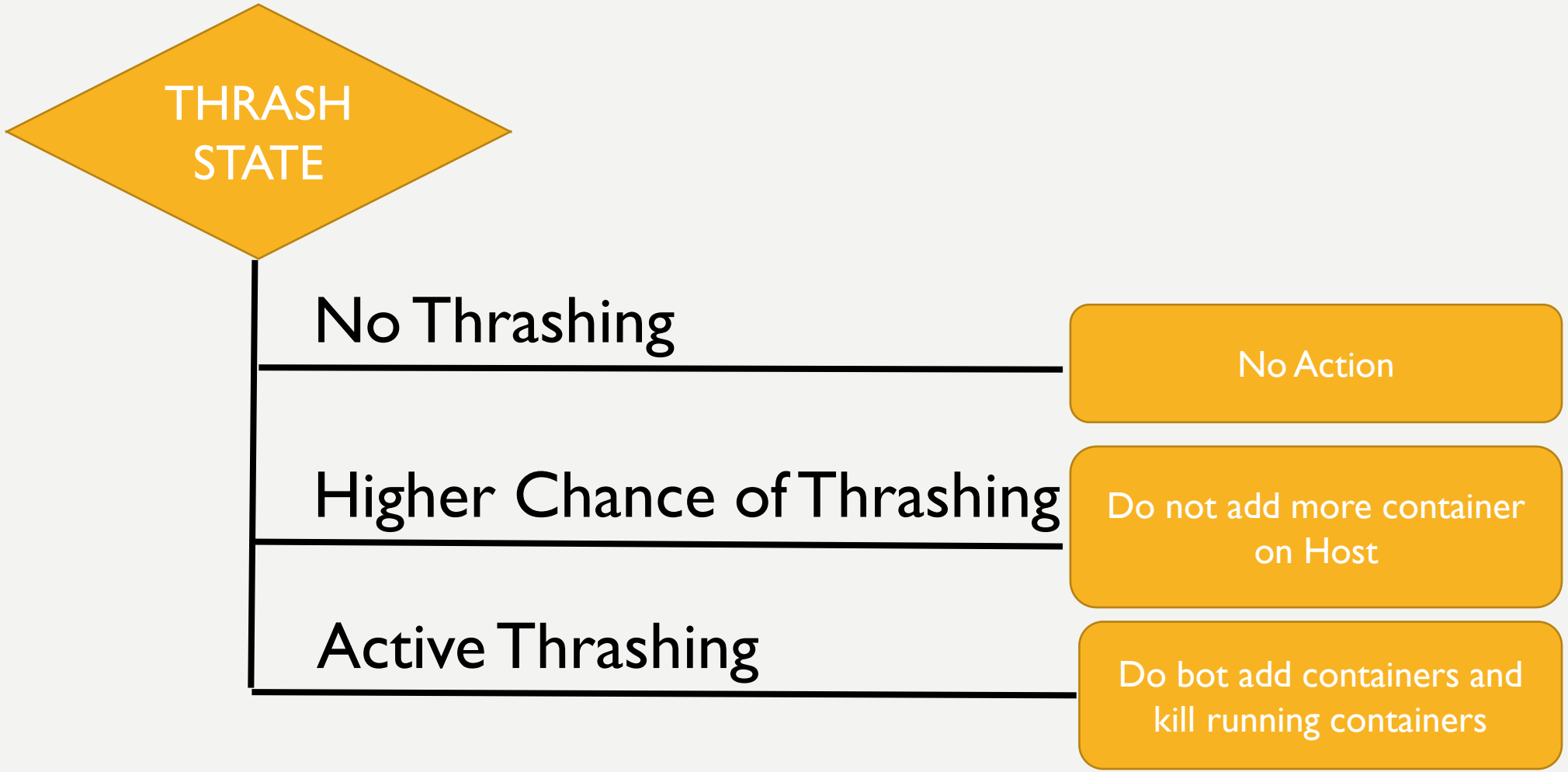
Score 1: Error in Thrashing Detection (detect thresholds thrashing event)

Score 2: Time to Detect Thrashing

(how quickly did we detect the event after it)

- *Note: Keep Score 1 lower than a threshold and then minimize Score 2*

ACTIONS TO TAKE:
SCHEDULER DECISION PROCESS



What Containers to Kill & What Not!

[Process Rule to adopt]

Rule 1 to follow the steps:

Step 1: Try to kill a Mapper
Step 2: Try to kill a Reducer
Step 3: Try to kill a Executor

Rule 2:

Never ever Kill Application Master

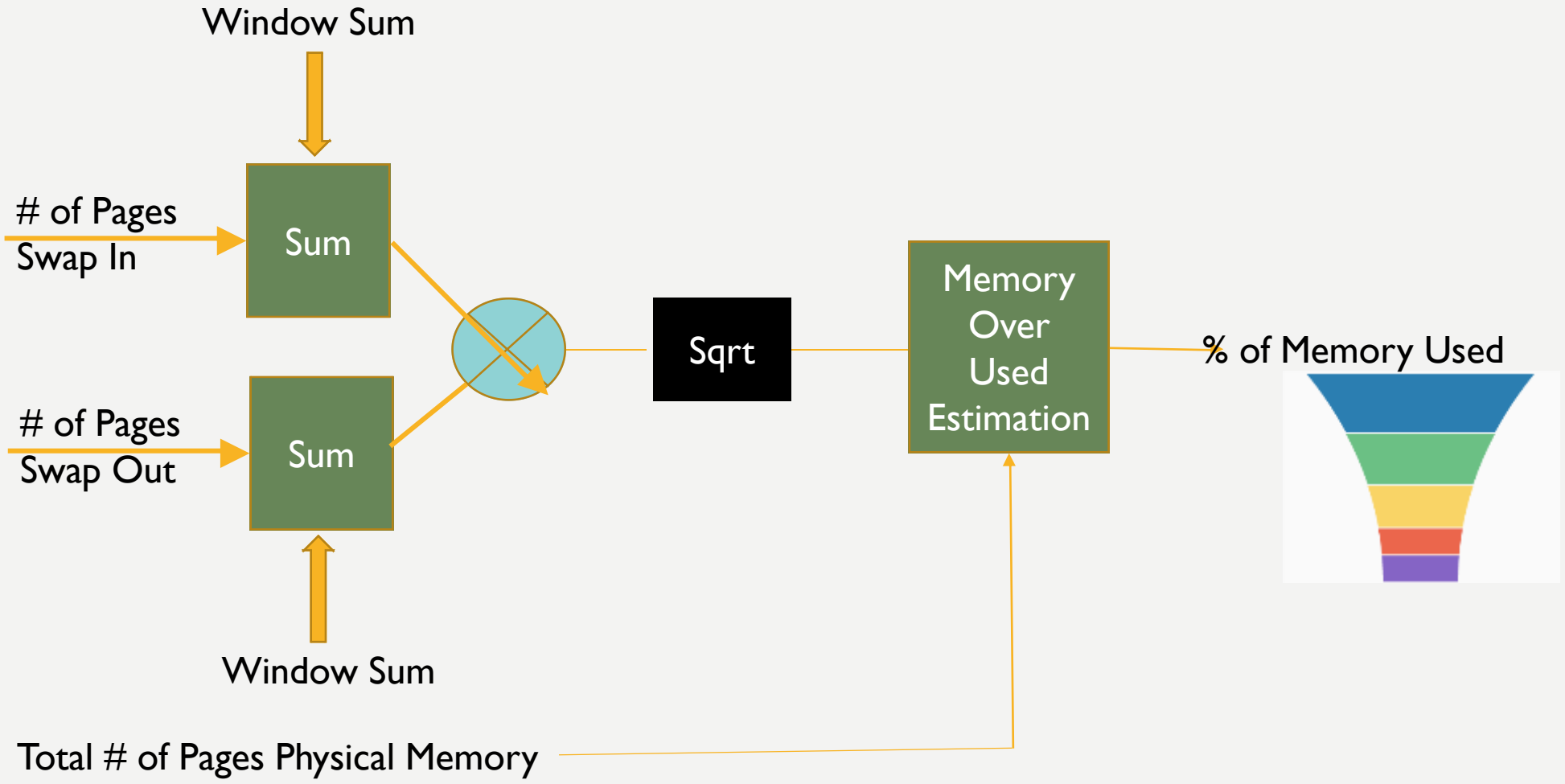
Dynamic Approach

1. Over allocation of Physical memory can lead to swapping thrashing
2. Aim to keep high throughout without thrashing
3. Need: Run less containers when thrashing is lightly
4. Difficult Need: Detect & Act very quickly (time in seconds)

What can we do!

A. Use Swap Space
B. Ensure Swap > 0
C. Be Conservative max RAM Usage on Node(s)

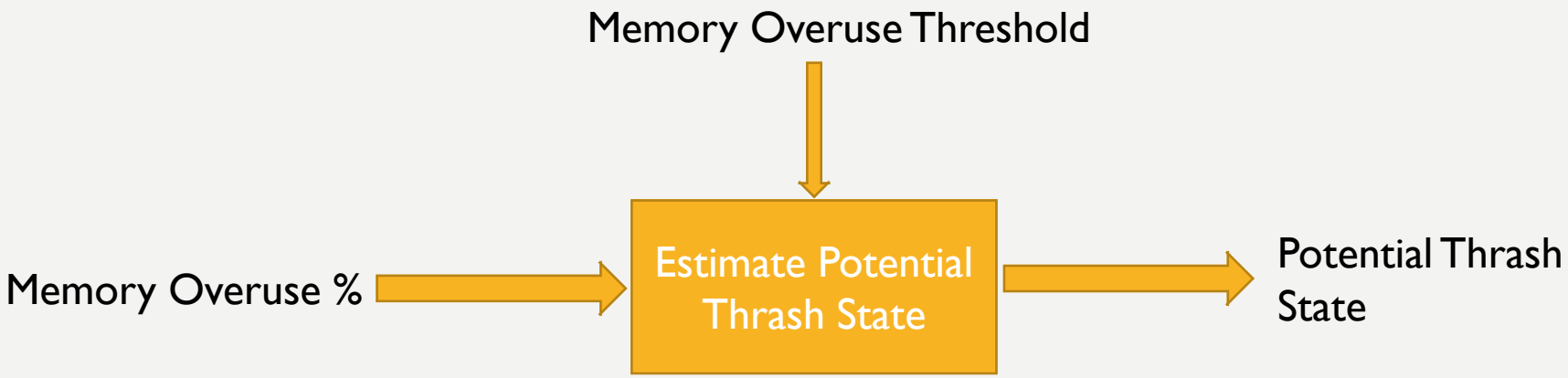
Memory Overload Estimation



Continue Pages Coming In * Out gives good estimate if extra memory usage

a. Overused Memory = geom_mean(SwapInRollSum, SwapOutRollSum)
b. Use this technique geometric mean → Check the o/p is high when both i/ps are HIGH
   And Output's units are pages of memory

MEMORY OVERUSE THRESHOLD



Potential Thrash State = Max (float (MemoryOverused%/MemoryOverusedThreshold), 2)

= 0
= 1
= 2

REFERENCES

1 Hadoop Echo Systems and Journals

2 G H FINE et al

Dynamic program behavior under paging

Proc 21 Nat 1 Conf.ACM IH66

3 LA BELADY

A study of replacement algorithms for virtual-storage computers

IBM Systems Journal 5 21966

4 Zen of Hybrid Modeling by Behera, R

Thank you!!