# DAMON: Kernel Subsystem for Data Access Monitoring and Access-aware System Operations

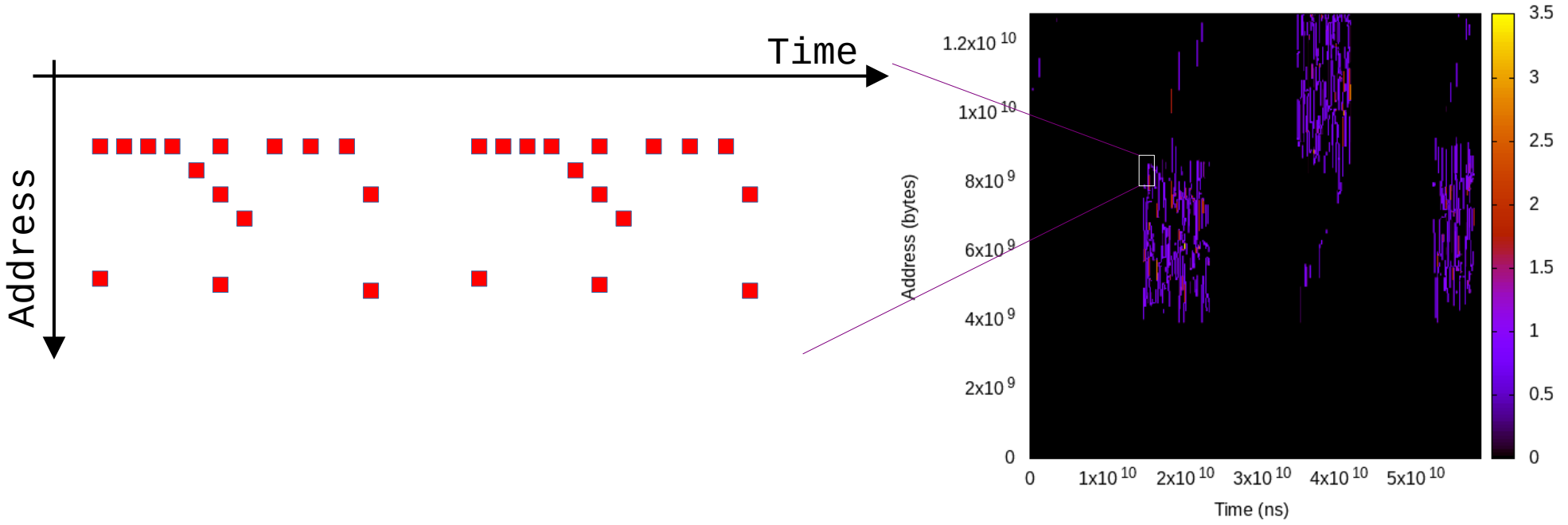
SeongJae Park (SJ) <sj@kernel.org> <sjpark@meta.com>

https://damonitor.github.io

QR code generated from https://qr.io

# DAMON: Kernel Subsystem for Data Access Monitoring and Access-aware System Operations

# Data Accesses: Events on Space/Time of Memory



# Data Access Monitoring: Hope, Real, and DAMON

- Hope: Precise, Complete, Light
  - Time granularity: CPU cycle / # CPUs (or, speed of light)
  - Space granularity: bit (or, electron)
  - Keep complete history (from Bigbang)
  - Lightweight enough to run on production systems
- Real: Expensive, YAGNI without tradeoff
  - O(memory size) time, O(memory size * total monitoring time) space overhead
  - Do you really care if a bit was accessed five years ago?
- DAMON: accuracy vs overhead tradeoff
  - Scalable, best-effort, aim real-world products usage

# Region-based Space Handling

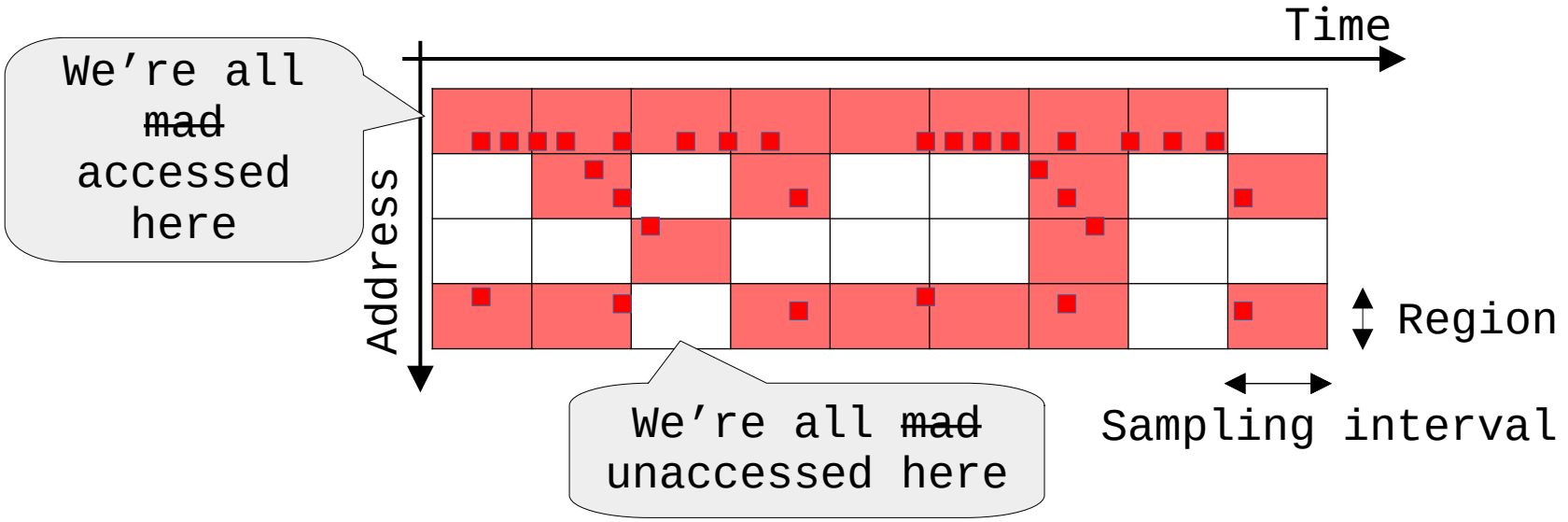
# Region: Access Monitoring Unit

- Defined as
  - A sub-area of the memory's space-time
  - A collection of adjacent elements that having similar access pattern
- Access check of one element per region is enough
- e.g., "This page is accessed within last 1 second; a cacheline is checked"

```
$ cat wonder_region_1
We're all ~~mad~~ [un]accessed here
```

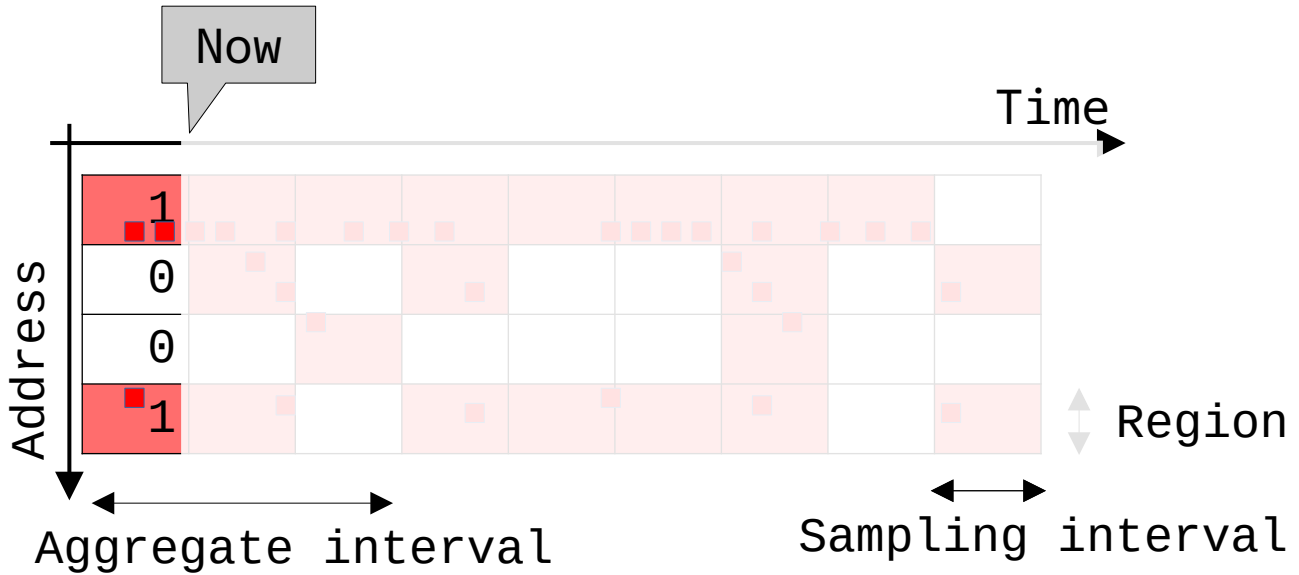
# Fixed Space/Time Granularity, <=1 Access Frequency

- Sort of periodic fixed granularity idleness monitoring
- Time overhead: “memory size / space granularity”
- Space overhead: “time overhead * monitoring time / time granularity”
- Reduced, but still ruled by memory size and total monitoring time



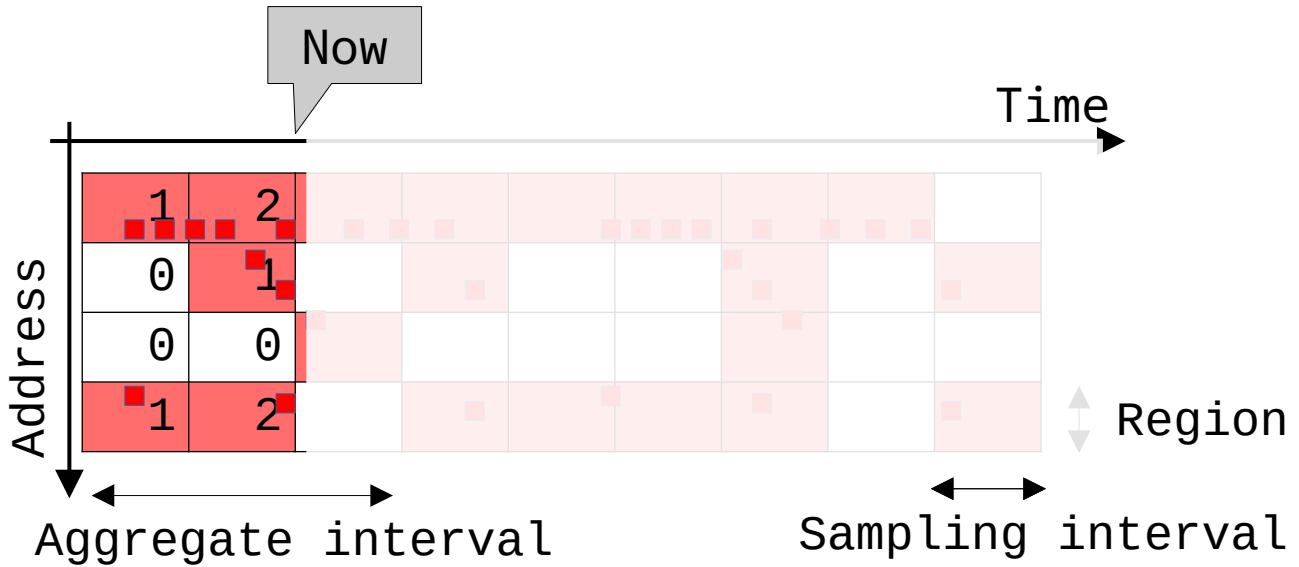
# Fixed Space/Time Granularity, <=N Access Frequency

- Accumulate access checks via per-region counter



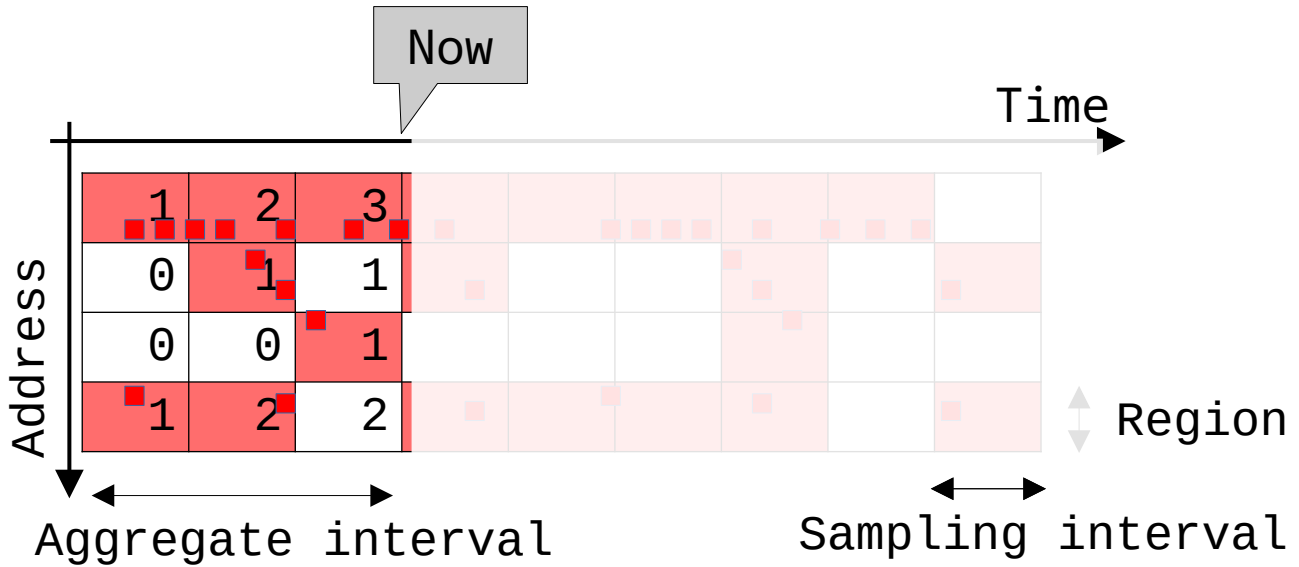
# Fixed Space/Time Granularity, <=N Access Frequency

- Accumulate access checks via per-region counter



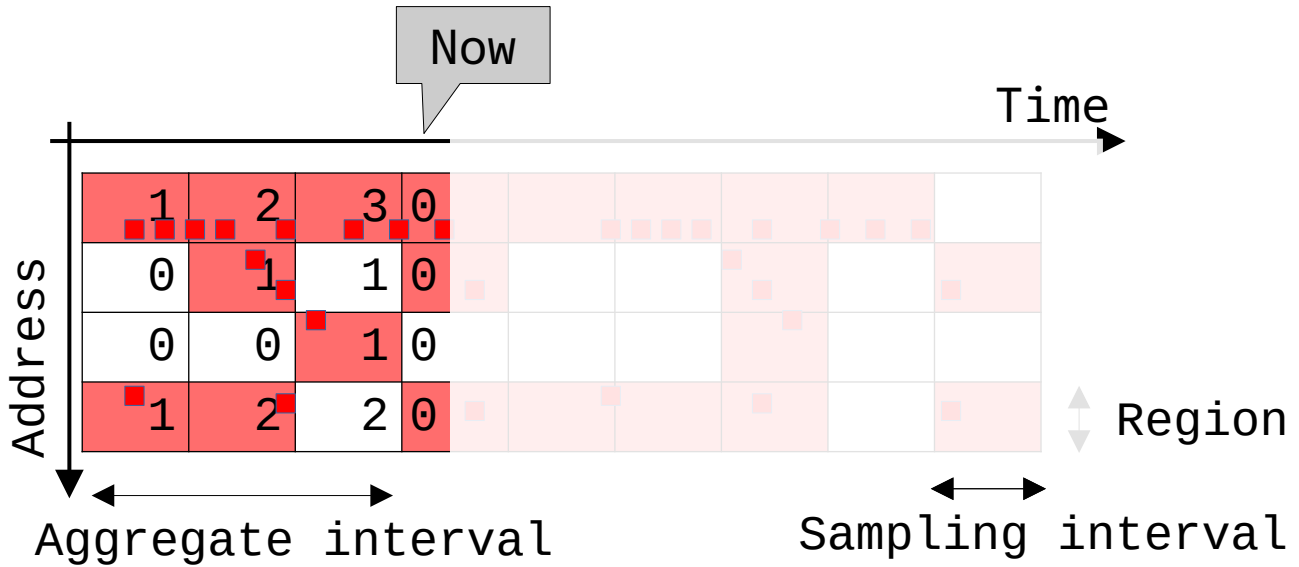
# Fixed Space/Time Granularity, <=N Access Frequency

- Accumulate access checks via per-region counter



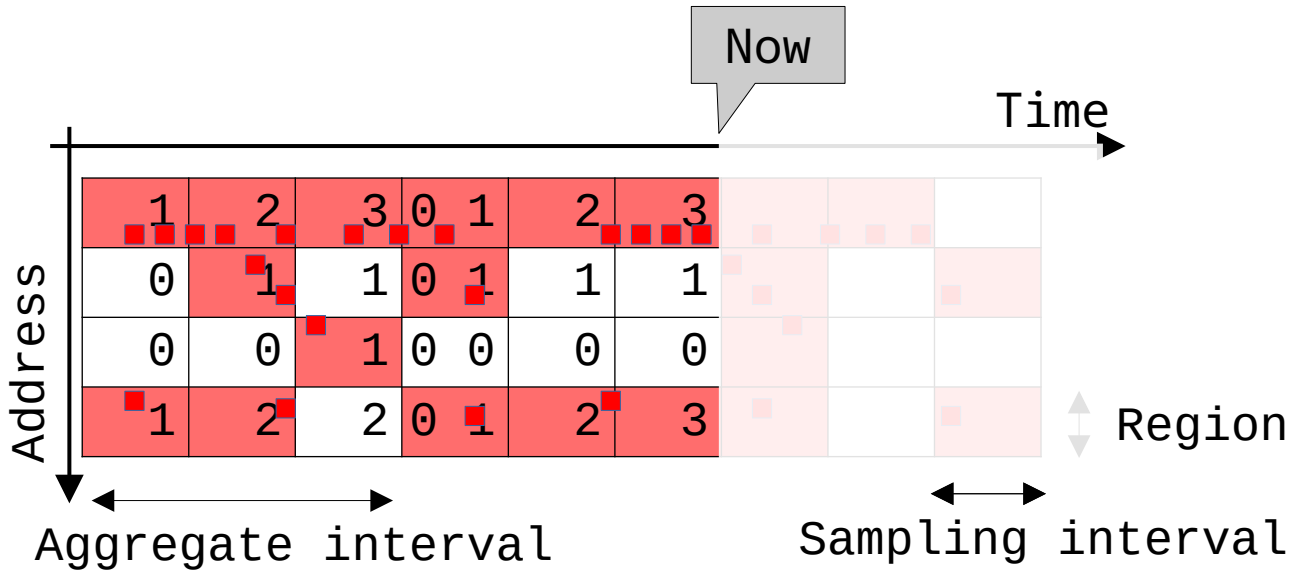
# Fixed Space/Time Granularity, <=N Access Frequency

- Accumulate access checks via per-region counter



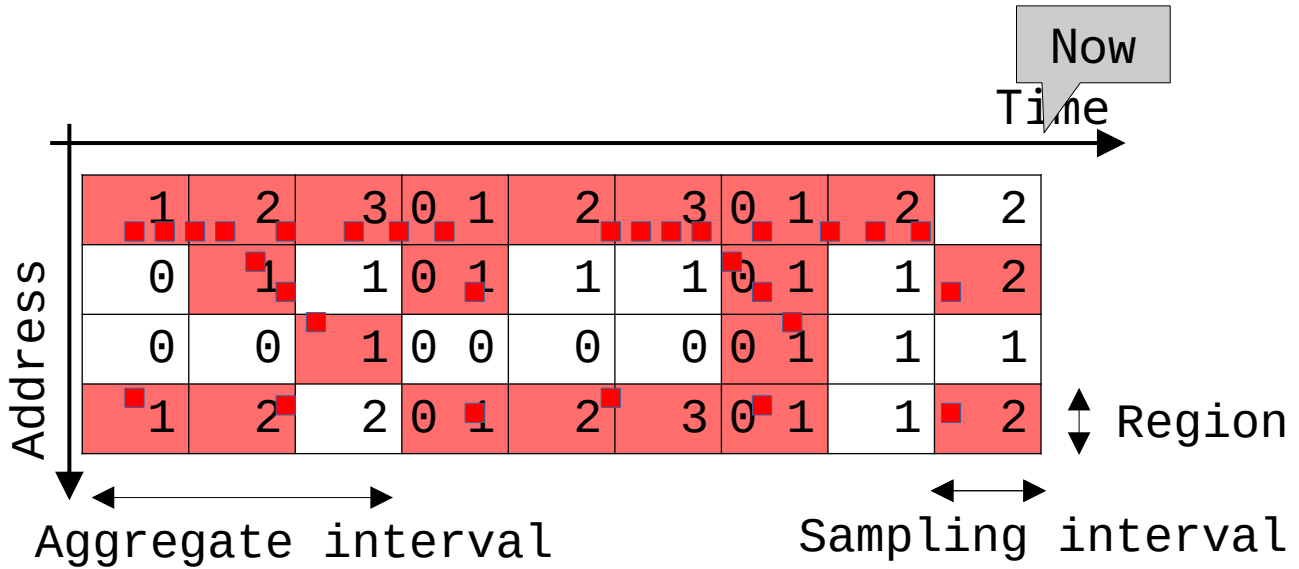
# Fixed Space/Time Granularity, <=N Access Frequency

- Accumulate access checks via per-region counter



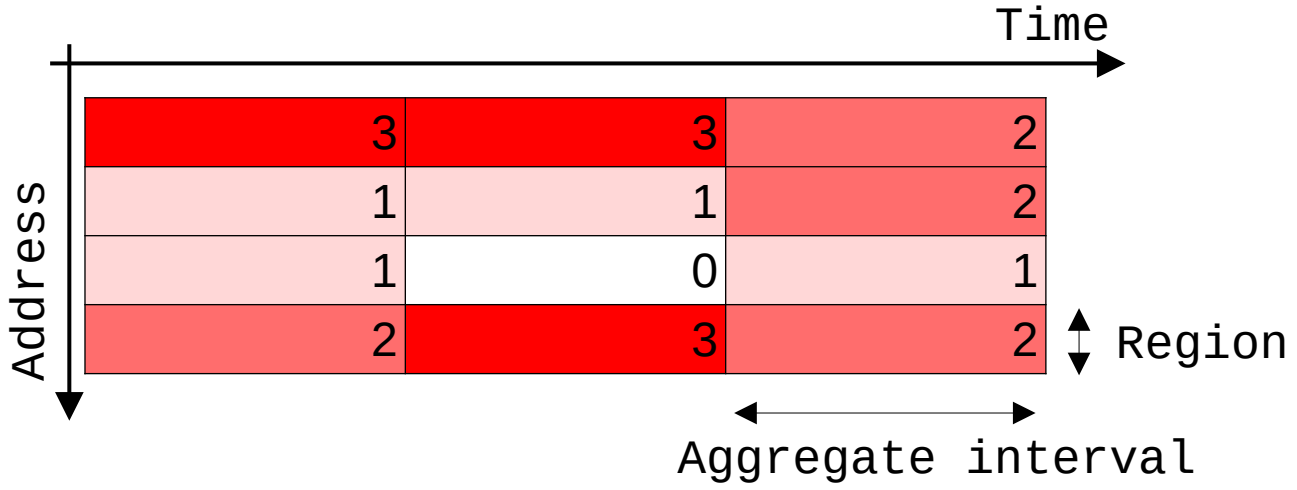
# Fixed Space/Time Granularity, <=N Access Frequency

- Accumulate access checks via per-region counter



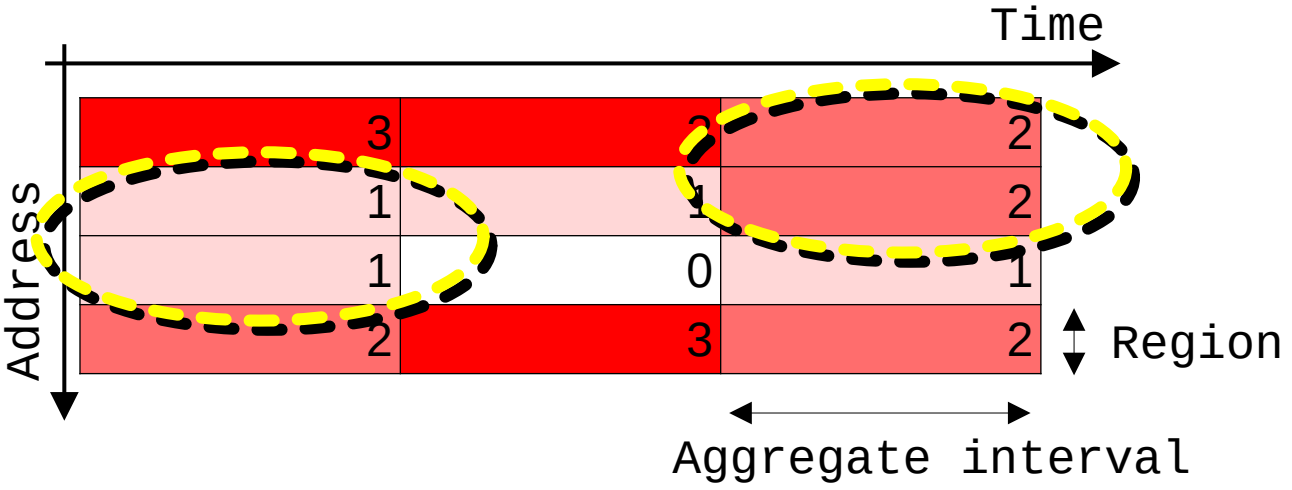
# Fixed Space/Time Granularity, <=N Access Frequency

- Accumulate access checks via per-region counter
- Reduce space overhead to “1/N”
- Still, O(memory size * total monitoring time)



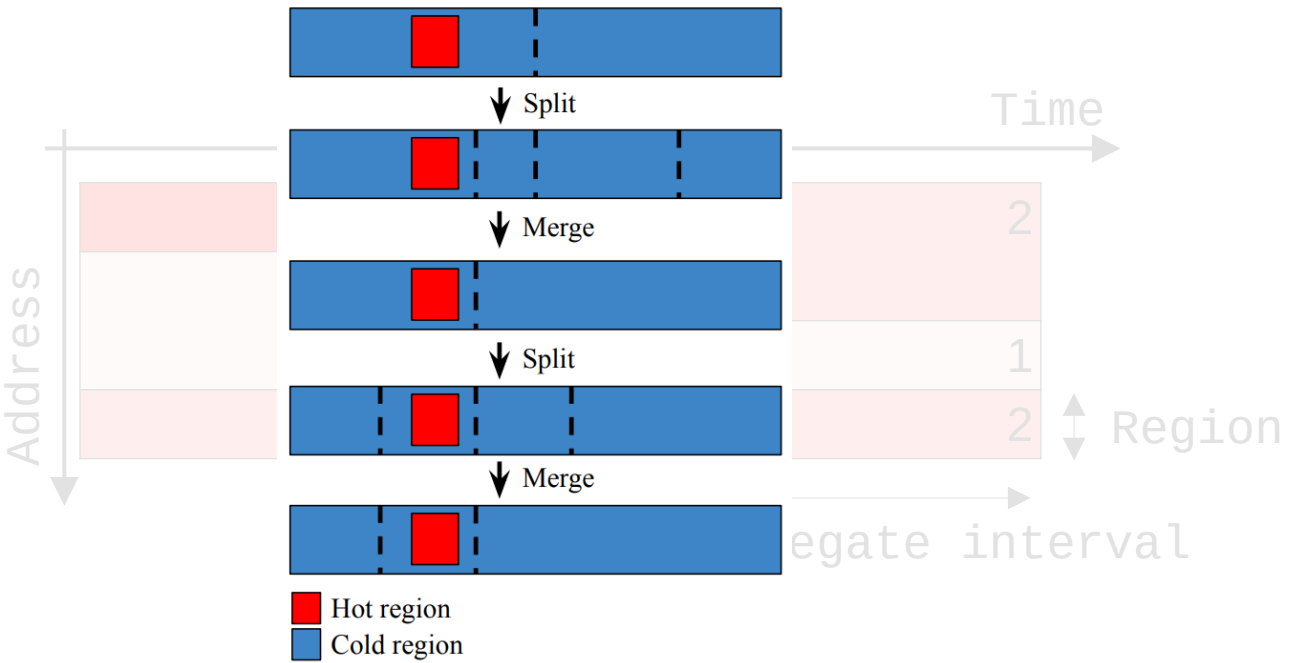
# Inefficiency of Fixed Space Granularity

- Adjacent regions of similar hotness



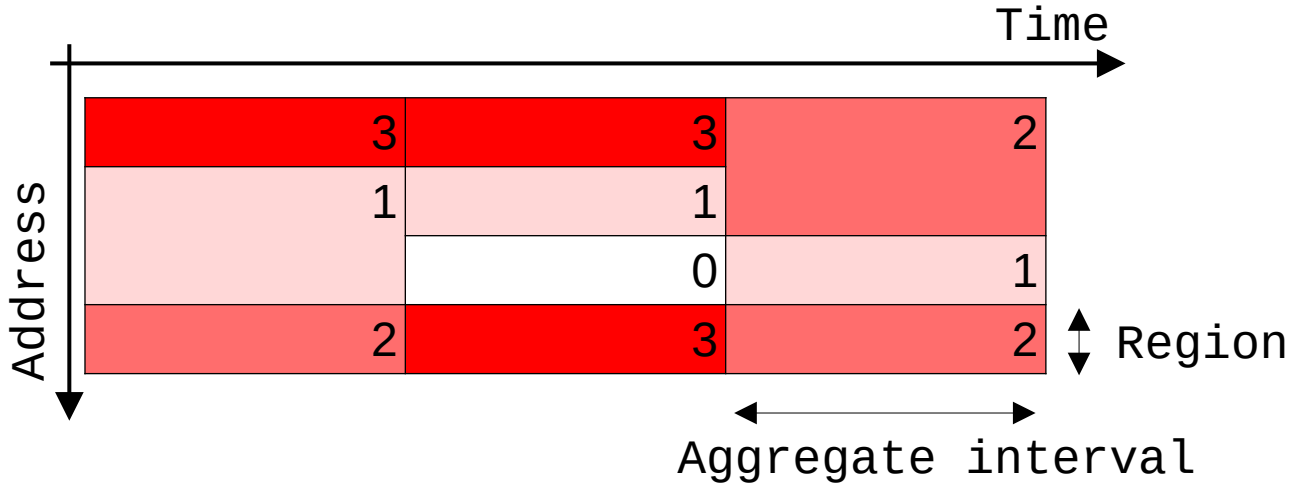
# Dynamic Space Granularity: Mechanisms

- Repeat merging the wasteful regions and randomly splitting regions
  - The number of region == number of different access patterns
- Let user set min/max number of total regions



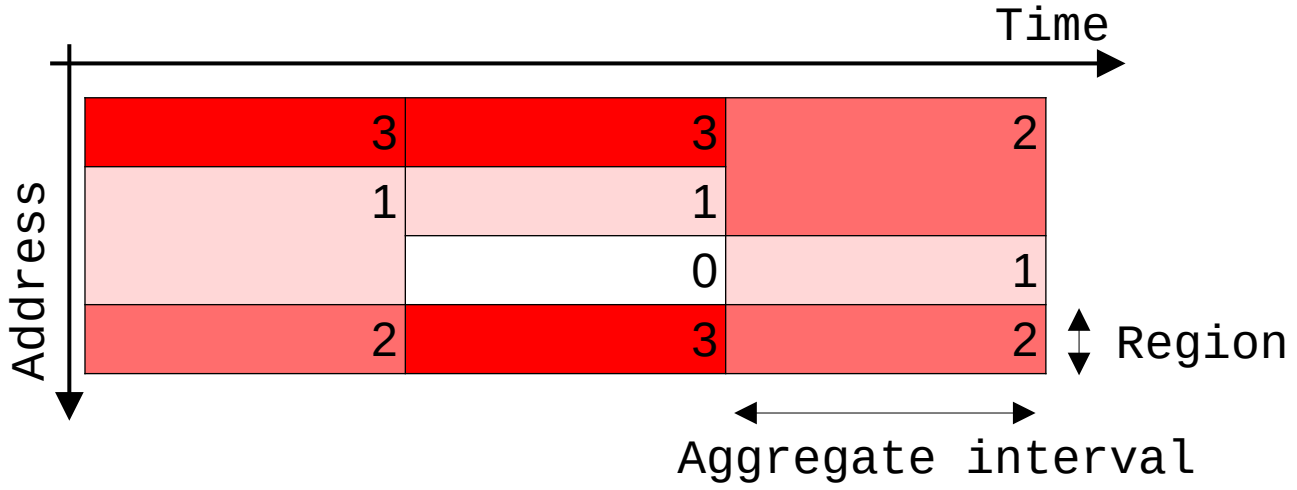
# Dynamic Space Granularity: Mechanisms

- Merge the wasteful regions and randomly split region
  - The number of region == number of different access patterns
- Let user set min/max number of regions



# Dynamic Space Granularity: Overhead/Accuracy

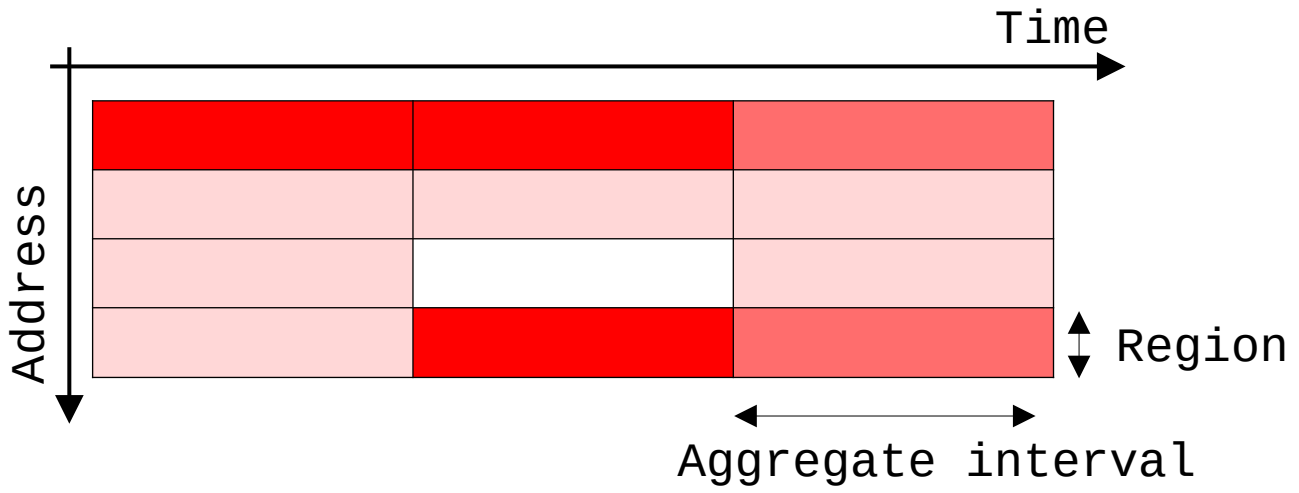
- Time overhead: min(different access patterns, max number of regions)
  - No more ruled by arbitrary memory size
- Accuracy: best-effort high
  - Lower-bound accuracy can be set via min_nr_regions



# Age Counting for History Handling

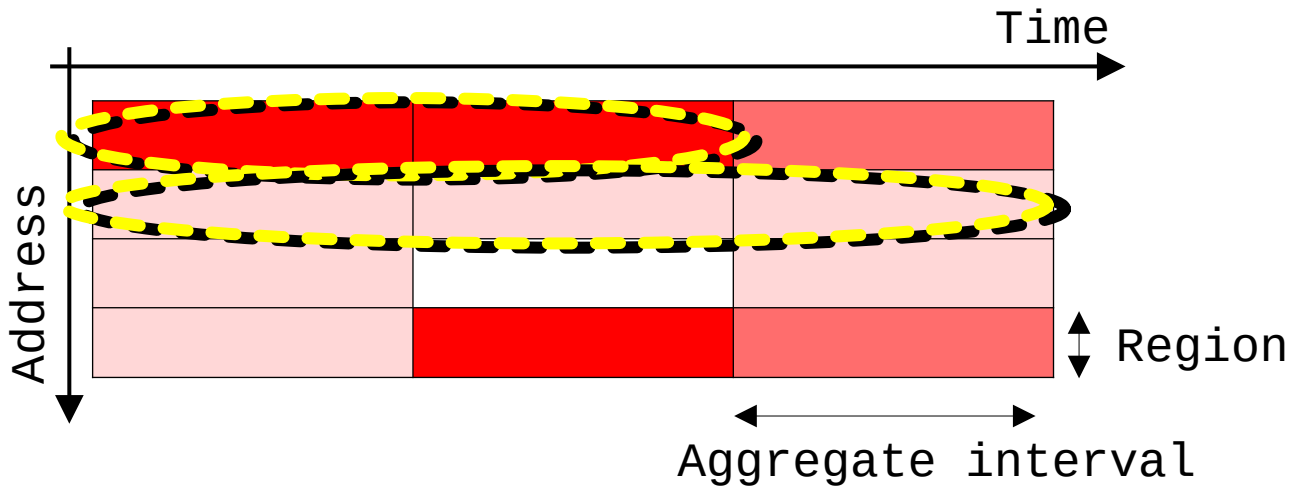
# Inefficiency of Fixed Time Granularity Regions (1/2)

- The definition of regions is not only about space, but also about time



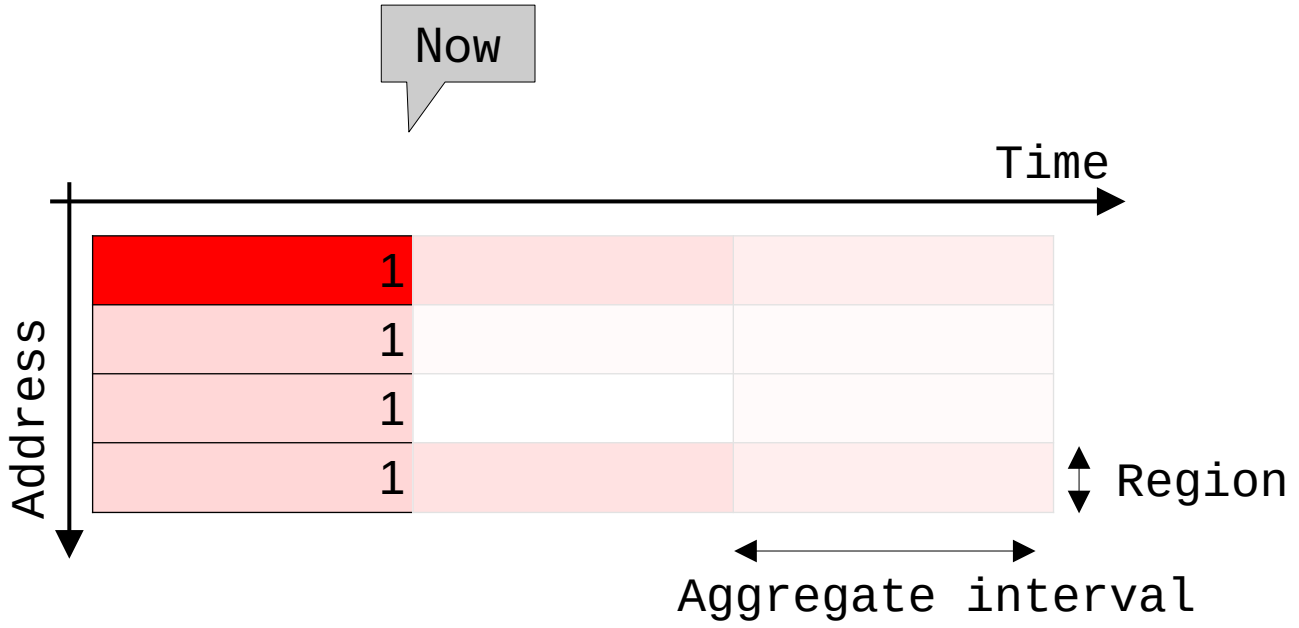
# Inefficiency of Fixed Time Granularity Regions (2/2)

- The definition of regions is not only about space, but also about time
- Multiple time-adjacent regions of similar hotness: only waste



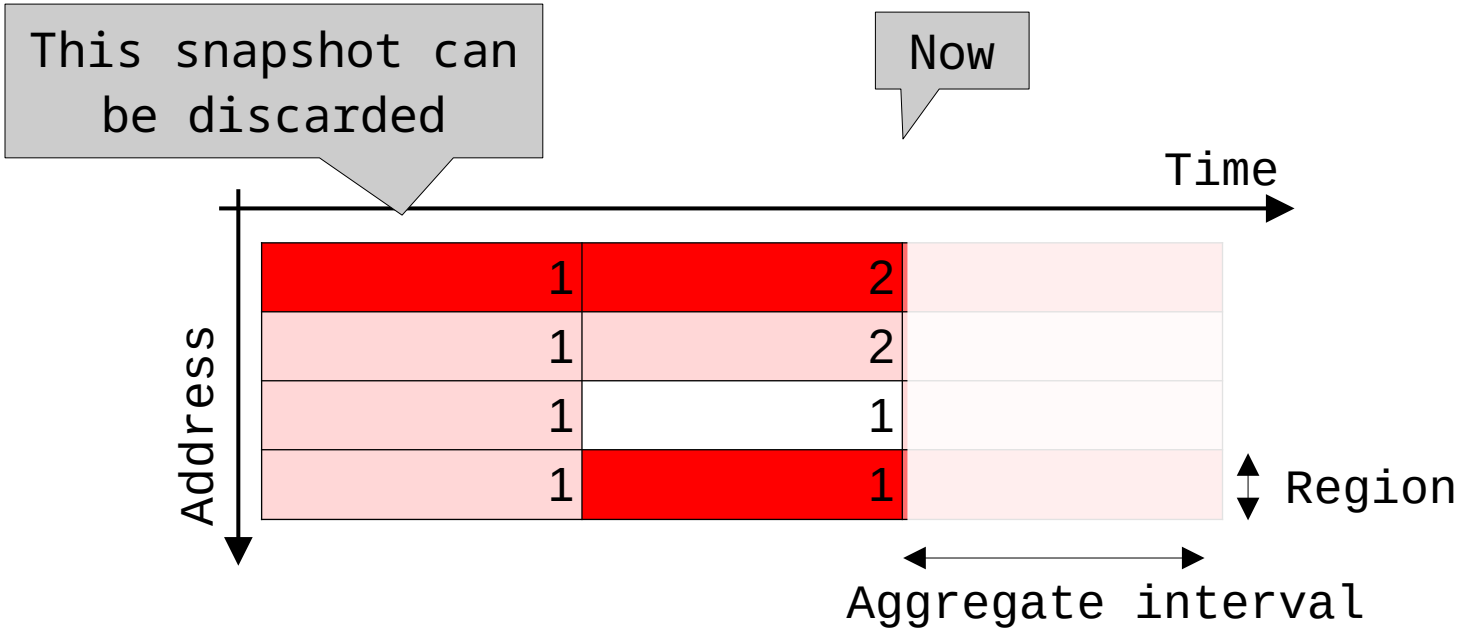
# Dynamic Time Granularity (1/3)

- Count how long the hotness has kept
- Snapshot contains history of useful length



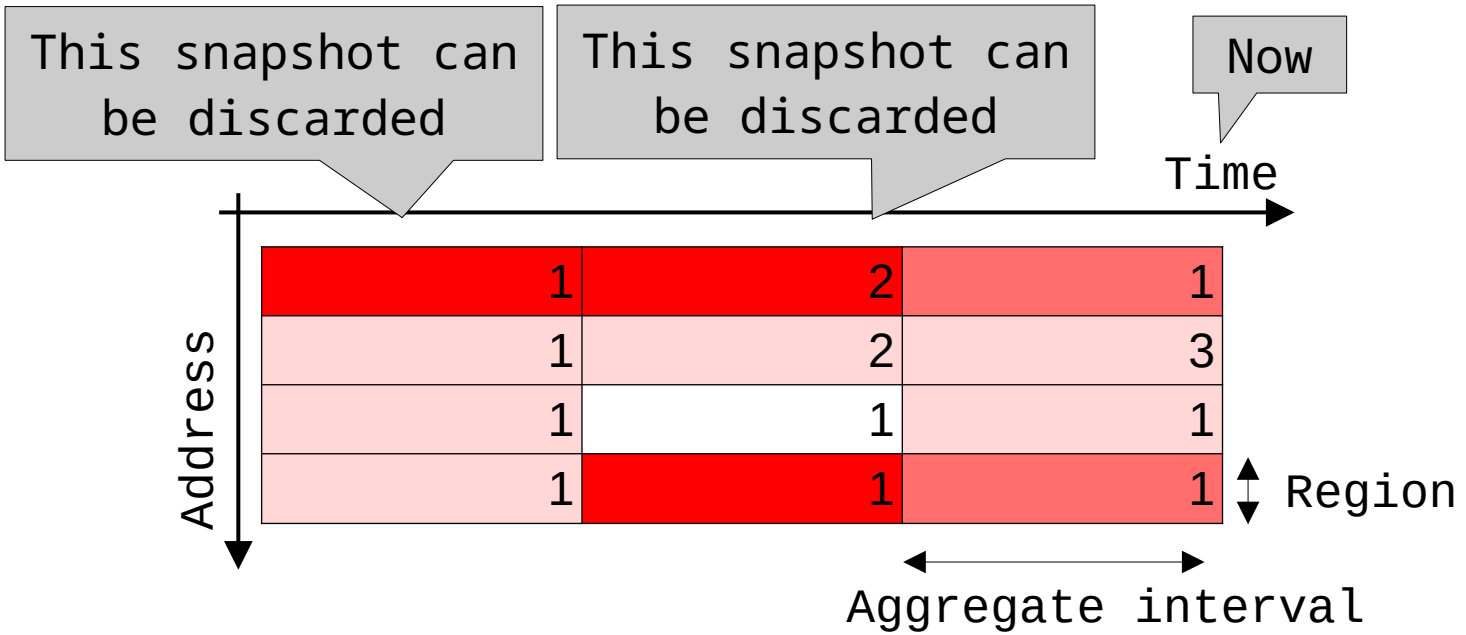
# Dynamic Time Granularity (2/3)

- Count how long the hotness has kept
- Snapshot contains history of useful length



# Dynamic Time Granularity (3/3)

- Count how long the hotness has kept
- Snapshot contains history of useful length



# Snapshot: The Output of DAMON

- O(max_nr_regions) time/space overhead
- Both time/space overheads are not ruled by memory size/monitoring time

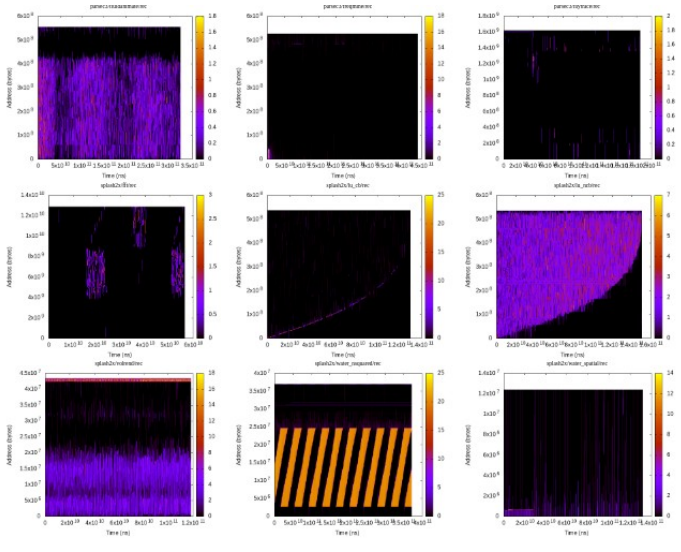
```
$ sudo ./damo report access --style simple-boxes
  |0000000000000000000000000000000000000000| size 53.340 MiB  access rate 0 %   age 58.300 s
|0000000000000000000000000000000000000000| size 53.020 MiB  access rate 0 %   age 1 m 16.800 s
 |0000000000000000000000000000000000000000| size 52.711 MiB  access rate 0 %   age 1 m 16.700 s
 |0000000000000000000000000000000000000000| size 53.289 MiB  access rate 0 %   age 1 m 16.500 s
 |0000000000000000000000000000000000000000| size 52.484 MiB  access rate 0 %   age 1 m 16.300 s
 |0000000000000000000000000000000000000000| size 52.887 MiB  access rate 0 %   age 1 m 15.800 s
 |0000000000000000000000000000000000000000| size 53.211 MiB  access rate 0 %   age 1 m 13.700 s
  |0000000000000000000000000000000000000000| size 52.777 MiB  access rate 0 %   age 59.200 s
            |00000000000000000000000000000| size 17.125 MiB  access rate 0 %   age 8.800 s
                                |55555555| size 8.000 KiB   access rate 60 %  age 400 ms
                     |9999999999999999999| size 7.672 MiB   access rate 100 % age 2.200 s
                     |8888888888888888888| size 1.922 MiB   access rate 95 %  age 2.200 s
              |0000000000000000000000000| size 53.121 MiB  access rate 0 %   age 7.200 s
              |0000000000000000000000000| size 23.238 MiB  access rate 0 %   age 8 s
    |0000000000000000000000000000000000000000| size 6.727 MiB   access rate 0 %   age 45.900 s
 |0000000000000000000000000000000000000000| size 124.000 KiB access rate 0 %   age 1 m 13 s
                                       |4| size 8.000 KiB   access rate 55 %  age 100 ms
total size: 533.660 MiB
```

# How Accurate, How Heavy?

- Up to you: Under the control, best effort (highest accuracy, lowest overhead)
- Being successfully used on real world products for years
- 3-4% single CPU usage is commonly reported from real world products
  - Many rooms to improve: the goal is turning it on by default everywhere

# Potential, or Aimed Usages

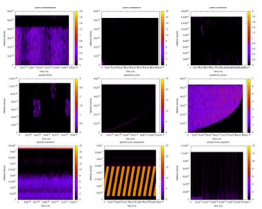
- Profiling (e.g., GIF demo link)
  - Help better understand and find rooms for improvements
- Profiling-guided Optimizations
  - Could be done on both offline and online
- Why not let kernel just (transparently) works?



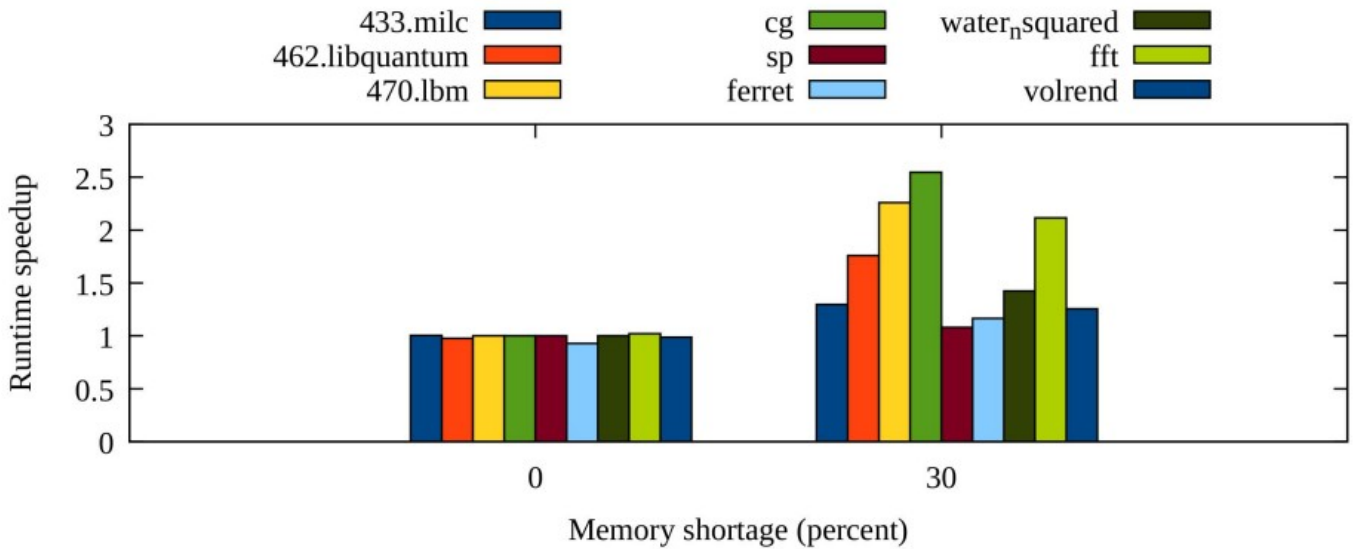
```
# Memory Footprints Distribution
percentile              0              25              50              75             100
      wss             0 B       9.520 MiB       9.543 MiB       9.785 MiB     107.039 MiB
      rss     104.820 MiB     104.820 MiB     104.820 MiB     104.820 MiB     104.820 MiB
      vsz     108.352 MiB     108.352 MiB     108.352 MiB     108.352 MiB     108.352 MiB
  sys_used       2.348 GiB       2.417 GiB       2.424 GiB       2.436 GiB       2.453 GiB

# Hotspot functions
# Samples: 589K of event 'cpu-clock:ppp'
# Event count (approx.): 147266750000
#
# Overhead  Command          Shared Object                       Symbol
# ........  ...............  ..................................  ..............................
#
    57.73%  swapper          [kernel.kallsyms]                   [k] pv_native_safe_halt
    40.26%  masim            masim                               [.] do_seq_wo
     0.11%  python3          python3.11                          [.] _PyEval_EvalFrameDefault
     0.09%  ps               [kernel.kallsyms]                   [k] do_syscall_64
     0.05%  ps               [kernel.kallsyms]                   [k] memset_orig
     0.04%  ps               libc.so.6                           [.] open64
```

# Potential, or Aimed Usages

- Profiling (e.g., GIF demo link)
  - Help better understand and find rooms for improvements
- Profiling-guided Optimizations
  - Could be done on both offline and online
- Why not let kernel just (transparently) works?

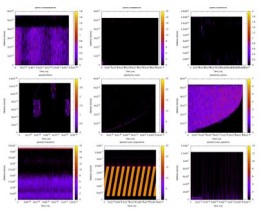


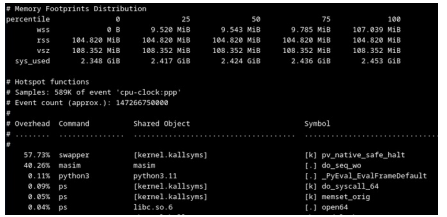
```
# Memory Footprints Distribution
percentile              0          25          50          75         100
       wss        0 B       9.520 MiB   9.543 MiB   9.785 MiB 107.039 MiB
       rss    184.820 MiB   184.820 MiB 184.820 MiB 184.820 MiB 184.820 MiB
       vsz    188.352 MiB   188.352 MiB 188.352 MiB 188.352 MiB 188.352 MiB
  sys_used      2.348 GiB     2.417 GiB   2.424 GiB   2.436 GiB   2.453 GiB

# Hotspot functions
# Samples: 589K of event 'cpu-clock:ppp'
# Event count (approx.): 147266750000
#
# Overhead  Command          Shared Object                      Symbol
# ........  ...............  .................................  ..............................
#
    57.73%  swapper          [kernel.kallsyms]                  [k] pv_native_safe_halt
    40.20%  masim            masim                              [.] do_seq_wo
     0.11%  python3          python3.11                         [.] _PyEval_EvalFrameDefault
     0.09%  ps               [kernel.kallsyms]                  [k] do_syscall_64
     0.05%  ps               [kernel.kallsyms]                  [k] memset_orig
     0.04%  ps               libc.so.6                          [.] open64
```

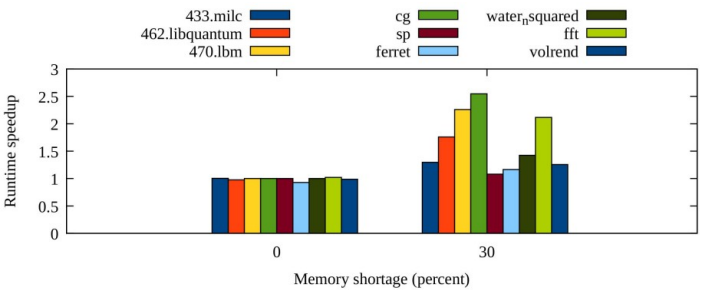


# Potential, or Aimed Usages

- Profiling (e.g., GIF demo link)
  - Help better understand and find rooms for improvements
- Profiling-guided Optimizations
  - Could be done on both offline and online
- Why not let kernel just (transparently) works?





# DAMON: Kernel Subsystem for Data Access Monitoring and Access-aware System Operations

# DAMOS: Data Access Monitoring-based Operation Schemes

- Another side of DAMON
- Let users define schemes
  - Memory operation actions to apply to regions of specific access pattern
- Once per user-defined time interval
  - find the regions of the condition from the snapshot and apply the action
- Additional features for production level control are provided
  - Finding optimum “access pattern” on dynamic environments is challenging

```
# # pageout memory regions that not accessed for >=5 seconds
# damo start --damos_action pageout --damos_access_rate 0% 0% --damos_age 5s max
```

# Features for Production Quality DAMOS Control

- Filters: define target memory with non-access-pattern information
  - “pageout cold pages *of NUMA node 1 that associated with cgroup A and file-backed*”
- Quota: set aggressiveness of DAMOS
  - Access pattern based prioritization is applied under the quota
  - “pageout cold pages *up to 100 MiB per second using <2% CPU time, coldest ones first*”
- Quota auto-tuning: auto-tune quota for user-defined target metric
  - “pageout cold pages *aiming 0.05% of memory pressure (coldest ones first)*”

# DAMOS for Efficient and Fine-grained Data Access Monitoring

- DAMOS_STAT
  - Special action making no system change but expose the scheme-internal information
  - Let user knows which of the memory are eligible for the scheme
- With DAMOS filters, can do page level properties-based monitoring
  - “How much of >2 minutes unaccessed memory are in hugepages and belong to cgroup A?”
- With DAMOS quotas, can do overhead-controlled monitoring

# Getting Started

# Availability

- Merged into the mainline from v5.15
- Backported and enabled on major Linux distro kernels
  - Major distros: Amazon, Android, Arch, Debian, Fedora, Oracle, …
- DAMON user-space tool (damo) is available on major packaging systems
  - Arch, Debian, Fedora, PyPi, ...

# How can I Use DAMON?

- Kernel API: Recommended for kernel programmers
- DAMON sysfs interface: Recommended for user-space program development
- DAMON user-space tool: Recommended for general usages from user-space
- DAMON modules: Recommended for specific usages
- If you don’t know what to do first, DAMON tests suite

# Community: For Questions, Help, Patch Reviews

- Public mailing list (https://lore.kernel.org/damon)
- Bi-weekly virtual meetup
  - Occasional/regular private meetings on demand
- Project website (https://damonitor.github.io)
  - Starting point for DAMON users and developers
- The future of DAMON is open and up to you
  - “Prefer random evolution over intelligent design”

# Summary: That’s DAMON

- DAMON is a kernel subsystem
  - For data access monitoring and access-aware system operations
- The future is open and up to the community
  - Make your selfish voice

# Questions?

- You can also ask questions anytime to
  - sj@kernel.org
  - Public mailing list (https://lore.kernel.org/damon)
  - Bi-weekly virtual meetup
  - Occasional/regular private meetings on demand
  - Project website (https://damonitor.github.io)

# Backup Slides

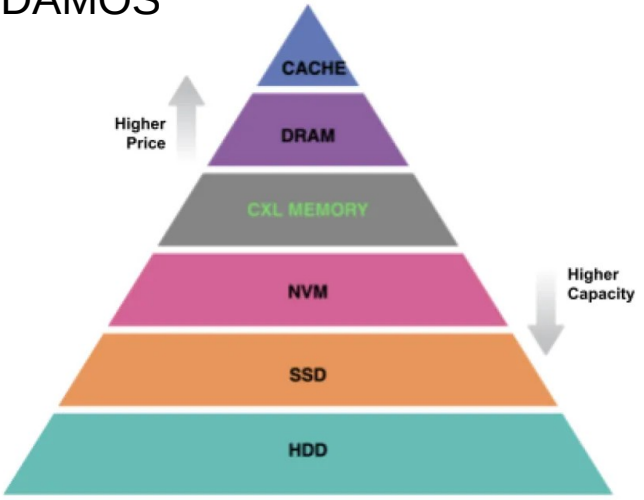
# Real-world DAMON Use Cases: Proactive Reclamation and CXL Memory Tiering

# Proactive Reclamation

- Reactive reclamation: Reclaim cold memory when memory pressure happens
- Proactively reclamation: Reclaim cold memory before memory pressure
- Benefit 1: Reduce memory footprint without performance degradation
- Benefit 2: Minimize degradation from direct reclamation
- Known usages: Google, Meta, and Amazon
  - Each company uses its own implementation for its usage
- AWS uses DAMOS-based implementation since 2022

# CXL Memory Tiering

- CXL-tiered memory: Put CXL memory between DRAM and NVM
  - Pros: Higher capacity with lower price (higher efficiency)
- Challenge: Dynamic placement of pages (CXL mem is slower than DRAM)
- DAMON-based approach: Place hot pages on DRAM node, Place cold pages on CXL node
- SK hynix developed their CXL memory SDK (HMSDK) using DAMOS
  - Reports ~12.9% speed up

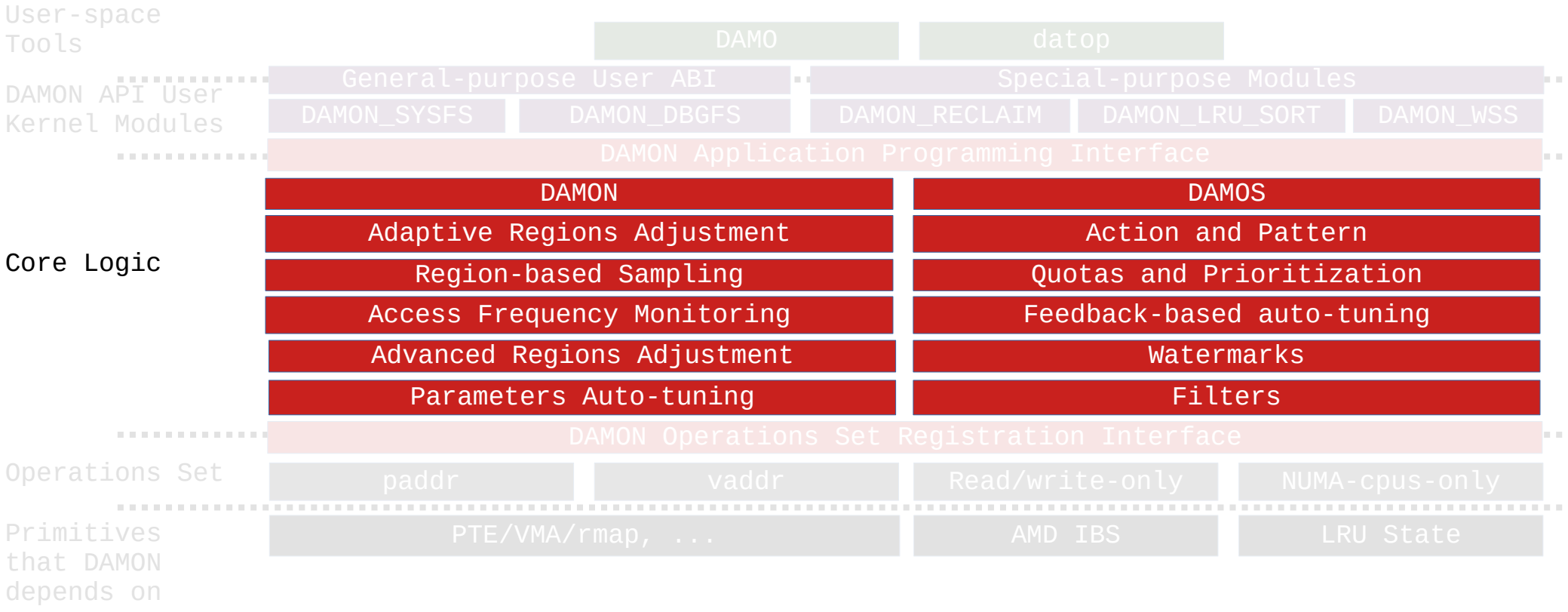


# Architectures

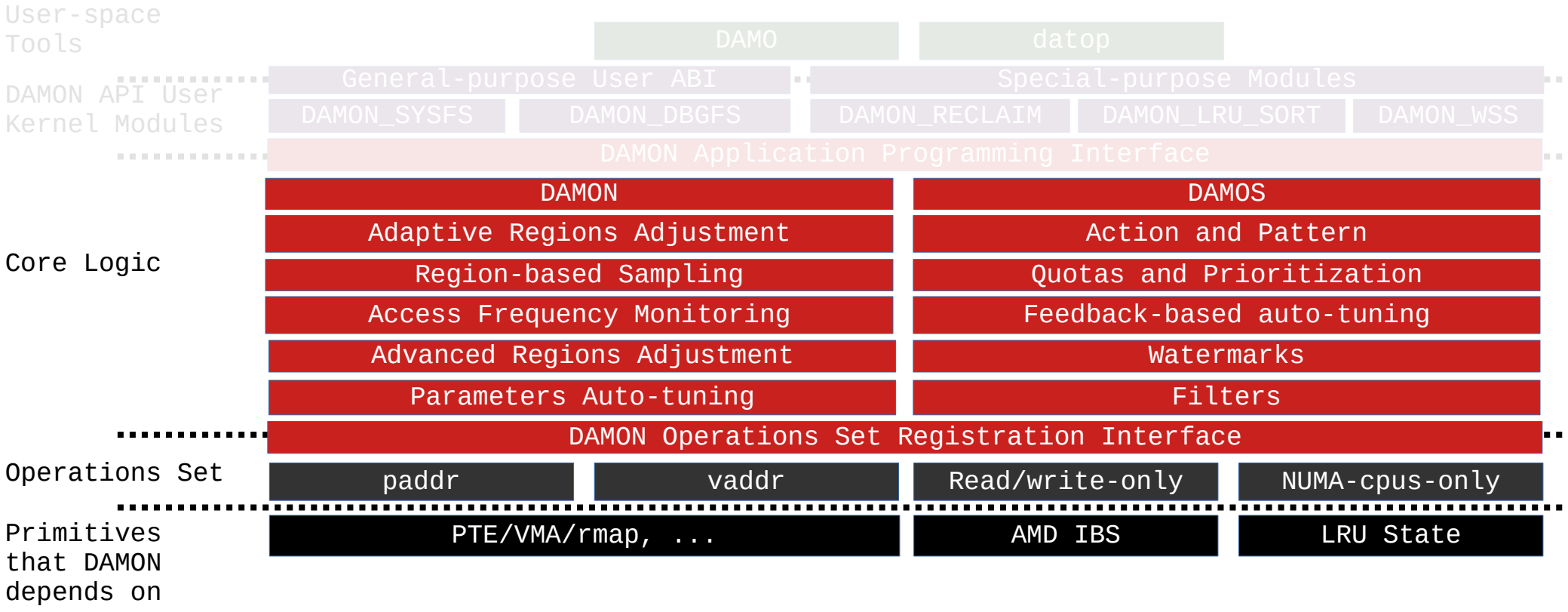
# Execution Model: Kernel Thread per Requests

- “struct damon_ctx”: Data structure for DAMON user input/output containing
  - User requests: target address space, address range, intervals, DAMOS schemes
  - Operation results: access snapshot, DAMOS stats
- “kdamond”: DAMON worker thread
  - Create one kdamond per “damon_ctx”
    - In future, could support multiple “damon_ctx” per kdamond
    - In future, could separate DAMOS to another thread (maybe useful for cgroup charging)
  - Allows async DAMON execution and multiple kdamonds (CPUs) scaling

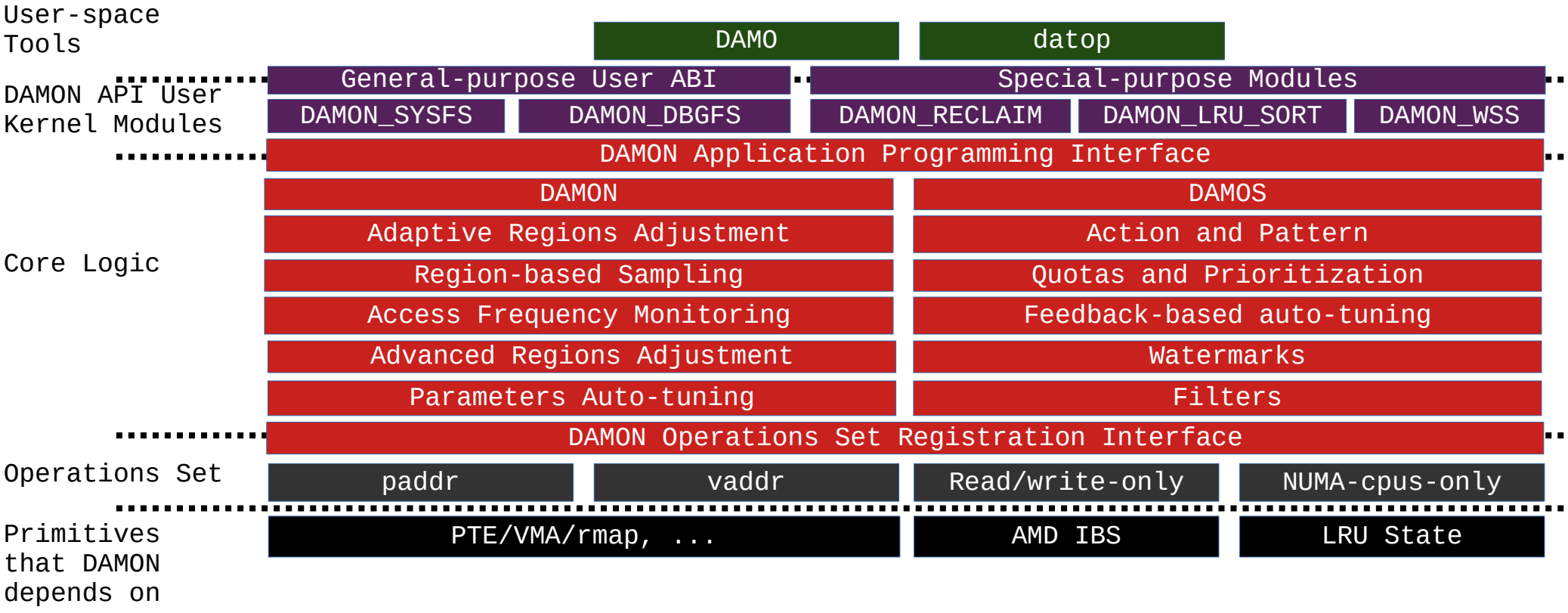
# Extensible Layers



# Extensible Layers



# Extensible Layers



# DAMOS Quotas: Intuitive Aggressiveness Control

- Before applying DAMOS schemes
  - Set temperature-based priority score of each region
  - Build “priority score”: “total size of regions of the priority” histogram
  - Find lowest priority threshold for the scheme meeting the quota
  - Skip applying action to regions having lower-than-threshold priority scores
- Single snapshot and histogram iteration: O(<=user-defined-N)
- Quota auto-tuning: A simple proportional feedback algorithm
  - Reward metrics: Arbitrary user-input or self-retrievable metrics like memory PSI

# DAMOS Filters: Fine-grained Target Selection

- Before applying DAMOS action, check the properties of region and skip action if needed
- Non-page granular (high level) filters
  - Filtered out before applying actions
  - Address ranges (e.g., NUMA nodes or Zone)
  - DAMON-defined monitoring target (e.g., process)
- Page granular (low level) filters
  - Filtered out in the middle of actions in page level
  - Anon/File-backed
  - Belonging memory cgroup
  - page_idle()

# Pseudo-code of DAMON v5.15

```
While True:
    for region in regions:
        if region.accessed():
            region.nr_accesses += 1
    sleep(sampling_interval)
    if now() % aggregation_interval:
        merge(regions)
        user_callback(regions)
        for region in regions:
            region.nr_accesses = 0
        split(regions)
```

# DAMON accuracy on Low-locality Space/Workloads

- It is proven to work on real world products for years
- Pareto principle and unconcious bias will make the pattern
  - Entropy-full situation is when the data center is doom-ed
- “age” avoid immature decision
- More works for accuracy improvement will be continued
- DAMON could be decoupled with the region-based mechanisms in future
- Let’s collect data and continue discussions together

# Can DAMON Extended for Non-snapshot Access Patterns?

- TL; DR: Yes, why not?
- DAMON is for any access information; Snapshot is one of the representations
- If the information/representation is useful for users, DAMON can add support
- We started discussion for Memory bandwidth visibility

# Can DAMON Use features Other than PTE Accessed bits?

- The extensible layer allows it
- AMD IBS and page fault-based appaoraches (e.g., PTE_NONE) are on the table
- In future, if GPU provides access check feature, we can extend to use it
- Such extension would allow
  - More lightweight and precise monitoring
  - Access source, read/write-aware monitroing
  - Kernel memory access monitoring

# DAMON Monitoring Auto-tuning

- Aggregation interval tuning is mandatory for real-world DAMON usages
- Auto-tuning for the user-desired amount of access samples is under development
- Auto-tuning of max_nr_regions for user-desired CPU usage is a TODO