Sampling Techniques

Alex Garthwaite

Monitor Group

December 19th, 2012

Sampling in VSphere

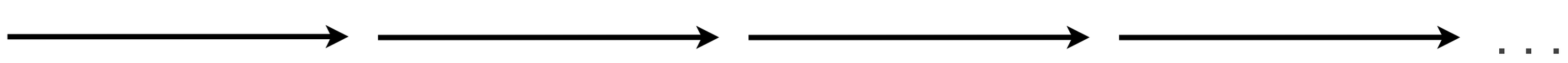
- **Simple wss estimates of mainMem at BPN level**
  - access: memory rebalance
  - dirty: vmotion estimates
- **Reasons to sample**
  - memory reclamation: comparitive measure of active
  - vmotion: estimate of dirtying rate/amount of work to do
  - VCOps, esxtop: measure of activity/efficiency of sizing of memory
  - DRS: anticipate expected memory need for placement decisions

Sampling in VSphere

- **Source code**
  - vmcore/vmm/main/busmemSample.c
  - vmcore/vmm/private/busmemSample.h
  - vmcore/vmx/main/busmemSample_user.c
- **Basic Framework**
  - Driven by 1Hz periodic callback
  - Based on vcpu progress (mean since vcpu0 skew)
  - Sample Period: 1 minute (ignoring first 4 seconds)
  - Sample History (number of sets): 4
  - Sample Size (per set): 100 pages
  - Each period:
    - update stats
    - kill old max-aged set
    - create new set of 100 pages
    - invalidate their mappings
  - VCPUs register touches when mainMem BPNs are translated to MPNs
- **Both vm configuration and VSI options to control settings**
  - needed? I think not

Sampling in VSphere

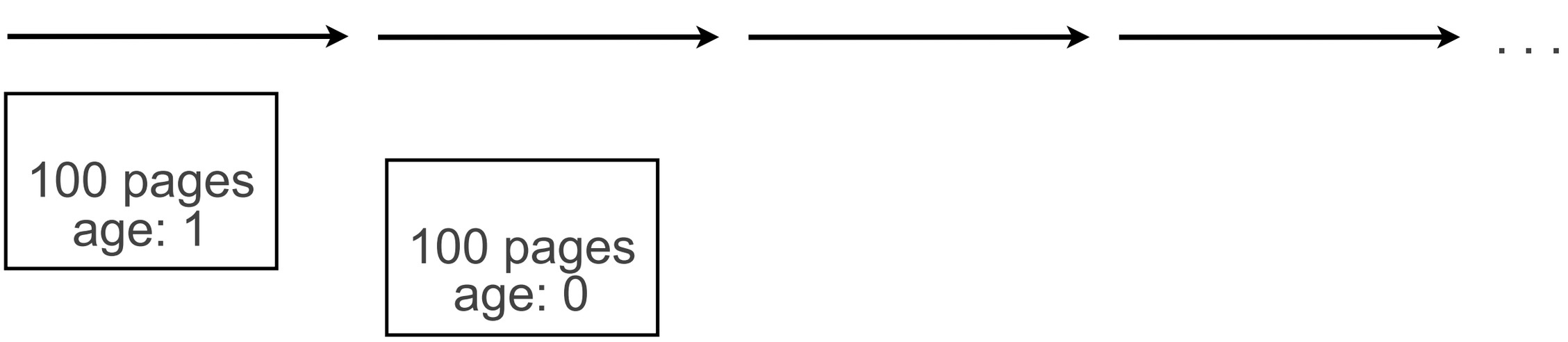
- **Start of Period**
  - concurrency
    - up to prod2013: stop all vcpus, use invalidation crosscall
    - now: use busmem lock and actions
  - choose 100 pages
  - invalidate mappings on all vcpus, even for pinned pages
    - first, global state: GPhys tables
    - in action, local state: MMU, etc.
    - leaves busmem frame/2MRegionInfo mapping state unchanged



100 pages
age: 0

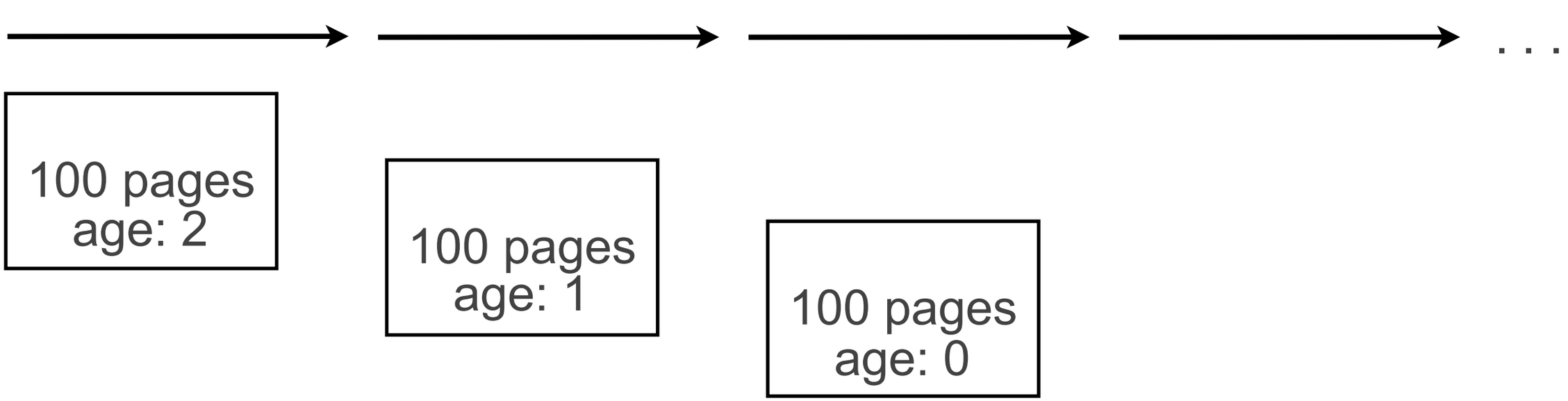
Sampling in VSphere

- **Updating statistics**
  - for each active set, compute estimates of pct touched, pct dirtied
  - for each age, aggregate fast and slow EWMAs
    - fast weights: 75%, 25%
    - slow weights: 25%, 75%
  - also: each (real) 1Hz Callback, measure fast/slow EWMAs for newest set
  - max of instantaneous, oldest, and youngest fast and slow averages used in memory balancing computations



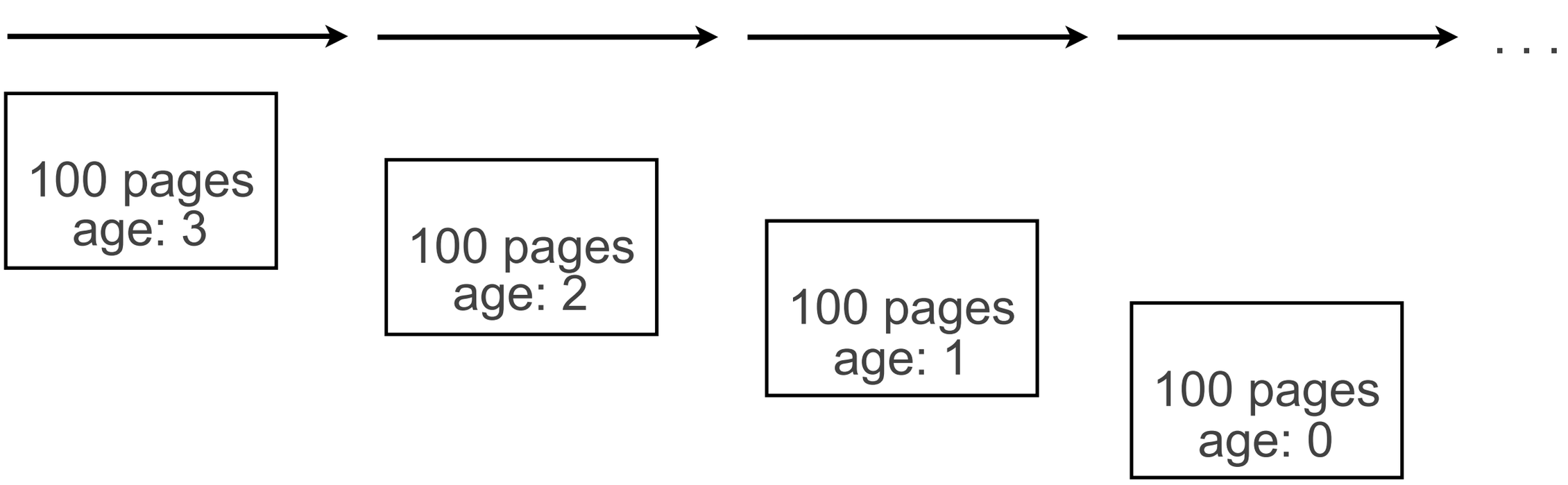
Sampling in VSphere

- **Ballooning**
  - may require guestOS to access/write sampled page
  - one might preserve activeness values when not ballooned
  - currently: clears tocuhed/dirty sampled state when ballooned



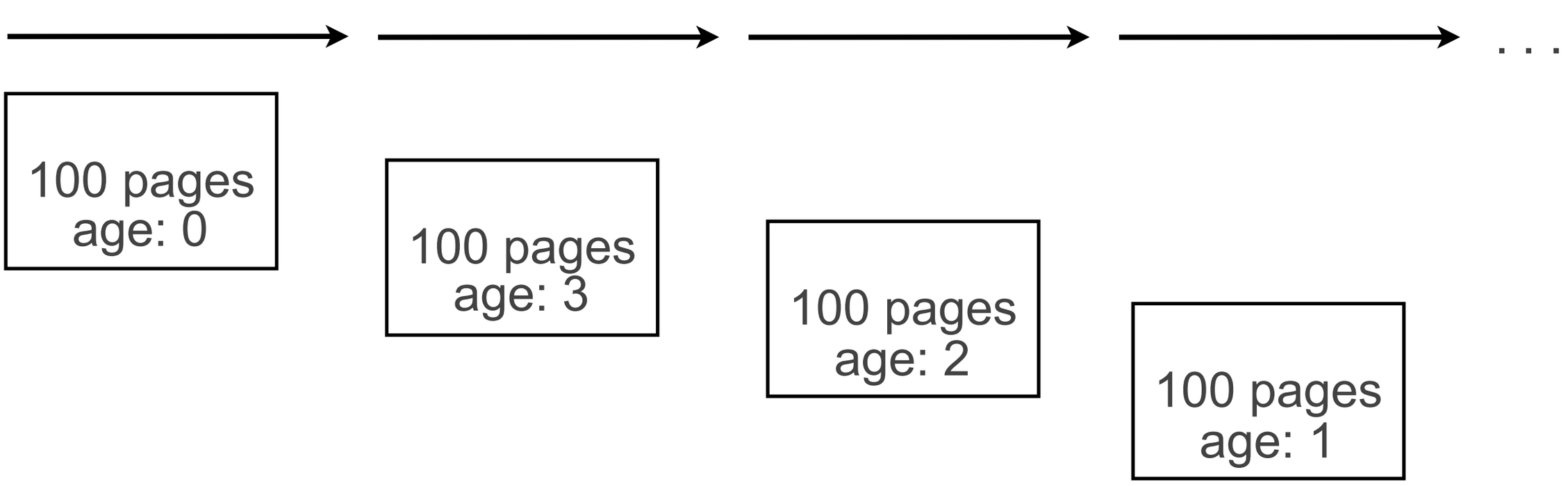
Sampling in VSphere

- **Invalidation**
  - under busmem lock, vcpu0 zaps GPhys for newly created set, posts actions
    - BusMemSampleInvalidateSampleSet, MONACTION_SAMPLE_SET, and BusMemSampleHandleInvalidations
  - each set tracks which vcpus need to issue local flush
  - in action handler under busmem lock, vcpus flush TLB and clear sw-mmu state, etc.



Sampling in VSphere

- **Special Cases**
  - (MMU) zero-pages considered touched/dirtied if selected
  - invalidation must be done at rank 0 (MonTLB, pinned pages)



Common Questions

- **Why fault-based?**
  - A/D bits used elsewhere: would need mechanism to share uses
  - A/D bits preset as optimization (e.g., TLB prefetching, PTE update costs)
- **What do instant, fast, slow EWMAs mean?**
  - relative value
  - unclear how each maps to a time-window
- **Why 4 sets of 100 pages?**
  - relative error is inverse of square root of sample size, not population
  - impact on large pages
- **Why rotate sample sets?**
  - avoid poor (fixed) choice of pages skewing analysis
- **Should we support sampling for VMs with full reservations?**
  - alternative: use guest-level measures of activity in guest (assumes trust)

Large Pages and Sampling

- **Large mappings count as touching 4KB pages**
  - inflates wss estimates especially when used aggressively
  - sched.mem.alwaysTryLPageAlloc
- **Large mappings held off in presence of sampling**
  - preventing large mappings hurts performance
  - 2MRegionInfo structure tracks number of failed attempts
  - allow large mapping when
    - no sampled 4KB pages remain or
    - failed-threshold exceeded (512 attempts)
    - Note: failed-threshold only reset when 2MRegion translated large(!)
      - need to make rate: recorded/reset each Period
- **Sample set size**
  - worst case: every sampled page falls on a different 2M region
  - 400 pages can block 800MB from being mapped large

Large Pages and Sampling

- **Identifying large-page behavior**
  - separate A-bit walk of GPhys tables: identify breadth of access
  - wss-sampling identifies “cold” 2MB regions
    - currently: revalidates in GPhys as large mapping
      - need to evaluate if this is useful since 2MRegion wasn’t accessed much
    - can use failed-attempt count to categorize 2MB regions
  - consider multi-level sampling
    - sample N large pages, M small pages within each
      - use M (possibly as fixed pattern) to capture breadth

Per-VCPU Sampling

- **Alex Milouchev's VI-A project: location-aware sampling**
  - per-VCPU stats for page accesses
  - rearm sample-sets every 10Hz
  - able to determine which VCPUs access pages on which NUMA nodes
- **Overheads**
  - rearming pages: 2400x over lifetime of a set
  - 2-vcpu SPECjbb VM: increased translation by factor of 64x
  - overheads increased: ~8% or ~110 seconds over run
    - VMK_GET_PFRAME_MPN: +~93.7 seconds
    - VMK_LOCK_PAGE: +~15 seconds
- BusMem_GetMPN
  - mapped bit still set for sampled pages but GPhys mappings gone
  - translation paths use BusMem_GetMPN
    - doubles #vmkcalls for sampled pages
  - Could fix BusMem_GetMPN to return INVALID_MPN for sampled pages
  - Could only call BusMem_GetMPN if page not destined for translation slow-path
  - Simplest is to just use BusMem_TryGetMPN

Latency-Sensitive VMs

- **Independently, Garrett found issues with the new barrier**
  - also saw high(er) BusMem_TryGetMPN costs
    - in BusMemInvalidateBackingStore
      - since most callers have zapped GPhys, unlikely to be useful
      - should just use INVALID_MPN in that code (especially for sampling)
  - saw serialization in the new action handler
    - Garrett's suggestion: post actions from vcpu to vcpu
    - alternative: don't require busmem lock in handler
      - only local state invalidated, similar to how zap-crosscall handler works
      - currently held to ensure set's BPN-list complete, for zero-page touching
      - could be done
  - should we consider not sampling highly latency-sensitive workloads
    - rely on guest metrics for right-sizing?

Miss-Ratio Curves and Marginal Benefits

- **Current metric too crude**
  - no notion of marginal benefit for adding or removing memory
  - idle-tax too coarse: leads to unfairness
- **Miss-Ratio Curves measure approx. fault-rate per allocation-level**
  - to contain cost, use combination of spatial and temporal sampling (RADIO paper)
  - track reuse-distances for pages in sample-set to build histogram
  - use histogram to compute number of accesses outside a given allocation level
- **Concerns**
  - overall cost of rearming every 10Hz
  - curve is flattened (especially toward the origin) through suppressing repeated accesses
  - access patterns within a period ordered by first access: slight lengthening of reuse-distances
  - sampling same # pages per VM or # pages proportional to memSize
    - how much each sampled page represents in comparisons
  - handling of resource pool hierarchy and aggregation: ignore for now
  - extend to, e.g., compression cache/swap to estimate larger allocations
    - unclear how to get feedback when ballooning
- **Should greatly improve memsched rebalancing**

Longer-Term Decisions: VCOps/DRS

- **Need longer term estimates of memory use**
  - current active estimate too crude/small window of time
  - needed for advising customers on right-sizing of VMs, placement
- **Rajesh's proposal**
  - consider possibly fixed set of pages
  - maintain reuse-distance histogram (64-bit counts)
  - each client can keep its own deltas for whatever time-scale it cares to track
  - need to checkpoint this histogram?

Current Status

- **Preeti moving sampling code to lib/sample**
  - plan is to move the sampling state to the platform
  - track in LockPage path when pages translated
    - can also capture vmkernel, vmx accesses
    - allows large sets of pages, also unifies sampling under memsched lock
- **Testing if in sample-set in vmm**
  - currently uses simple bloom filter to avoid updating any busmem state
  - Preeti's observation: clear dirty bits instead
    - busmem frames track dirtiness of 4KB pages: cache of platform dirtiness state
    - clearing this in vmm forces translations for write down slow-path (to LockPage)
    - would also need to switch to using BusMem_TryGetMPN to get MPN for non-write accesses
    - allows one to drop the bloom filter, simplify the translation slow-path test
- **PR957959 and sampling/GPhys interaction**
  - currently trying to verify if due to GPhys page-table walker taking "isLarge" flag

Recap

- **Fix sampling/GPhys bug (pr957959)**
- **Move sampling code to platform**
- **Implement/evaluate MRCs for memory balancing**
- **Improve sampling invalidation barrier**
  - eliminate use of busmem lock in handler
  - stop using BusMem_GetMPN in BusMem translation paths
  - stop using BusMem_TryGetMPN in BusMemInvalidateBackingStore
- **Use sampling activity to help categorize large-page use**
  - consider two-level selection at 2MB granularity
  - breadth, coldness measures
  - fix failed-attempt metric
  - consider dropping revalidation step for cold large pages
- **Consider not using sampling if latencies too high**