L07a. Global Memory Systems

Introduction:

- Global Memory System – GMS: How can we use peer memory for paging across LAN?
- Distributed Memory System – DMS: Can we make the cluster appear like a shared memory machine?
- Distributed File System – DFS: How to use cluster memory for cooperative caching of files?

Global Memory System and Cluster Memory:

- The Virtual Address Space of a process is much larger than the Physical Memory that is allocated for that particular process.
- The Working Set of a Process is defined as the portion of Virtual Address Space of the process that is actually present in Physical Memory.
  - The Virtual Memory Manager gives the illusion to the process that all of its Virtual Address Space is contained in Physical Memory by paging in and out from the disk the pages that are frequently accessed by the process. That is, the Working Set of a Process is always contained in Physical Memory.
- Memory pressure on a particular node is the amount of physical memory used to keep the working set of all the processes in physical memory.
  - If the Memory pressure is high, then the physical memory available is not sufficient to hold the Working Set of all processes in the physical memory.
  - When multiple nodes are connected on a LAN, the memory pressure on each node will be different since the characteristics of the processes and workload on different nodes will be different.
- If some nodes are idle and some nodes are loaded/busy, we can use the idle cluster memory of a peer node for paging in and out the working set of processes on a busy node.
  - The benefit of paging in and out to peer cluster memory is that accessing remote memory is faster than accessing local disk.
  - For instance, accessing a local disk can take around 10 milliseconds, whereas accessing remote memory can take around 100 microseconds.
  - If we use specialized network cards, accessing remote memory can take around 20 microseconds or far less time!
  - Accessing a spinning disk involves Seek Latency and Rotational Latency and so the overall transfer rates are around 200 MB/s. Accessing remote memory through Gigabit Ethernet cables can range from 100 MB/s to around 5 GB/s.
    - In summary,

How GMS uses Cluster Memory:

- GMS extends the normal memory hierarchy by adding the cluster memory as a paging option.
- In GMS, when a Page Fault happens, the GMS checks the remote cluster memory of peer nodes for the missing pages before checking the local disk, because accessing remote memory is faster than accessing local disk.
- In GMS, the only pages that can be in cluster memories are clean paged-out pages.
- When the GMS VMM decides to evict a page from physical memory to make room for the current working set of processes on a node, it checks the network for an idle peer node and puts that evicted page in peer memory for later retrieval.

GSM Basics:

- In GSM, Cache refers to Physical Memory (i.e. DRAM) and not CPU/processor L1/L2 Caches.
- A Page can have two states:
  - Shared Page: The same page is copied in the physical memory of multiple nodes and used by an application that spans across multiple nodes of a cluster (e.g. Oracle RAC application).
  - Private Page: A page is present only in the physical memory of a local node and is not present on any other cluster node since the application using that page runs only on that local node.
- Page Faults for a particular node are handled by the community of peer cluster nodes. Such that the memories of peer cluster nodes serve as supplement for local disk of the particular node.
- The Physical Memory of a particular node is split into two sections:
  - Local Memory: Contains the Working Set of local processes on that particular node. Local Memory contains Private Pages and/or Shared Pages.
  - Global Memory: Spare Memory for peer cluster nodes (community service by particular node). Global Memory contains only Private Pages swapped-out by peer cluster nodes on the network. Thus, if a page is in Global Memory, it is guaranteed to be Private and cannot be a Shared page.
- The memory pressure on a particular node is not constant and varies with time.
  - If the memory pressure on the particular node causes more local memory to be required to hold the working set of local processes, then the local memory portion increases.
  - On the other hand, if the node is idle, then the local memory portion may shrink, and the node can allow more of the peer nodes' swapped-out pages to be present in the global portion of the physical memory of that node, thereby increasing Global portion.
  - This boundary between Local and Global Memory portions is dynamic in response to memory pressure on that particular node
- In summary, the idea of GSM is to serve as a remote paging facility.
  - Maintaining the coherence of multiple copies of a page is not a GSM problem, it is an application problem.
  - GSM chooses a Globally Least Recently Used (Global LRU), which considers the oldest page on the whole cluster, as the page replacement policy.
  - GSM has to manage age information for the pages across all the nodes in the cluster.

Handling Page Faults – Case #1:

- What happens if a Page Fault happens on node P?
- Memory on hosts P and Q is split into Local Memory and Global Memory each.
- If a Page Fault for Page X happens on node P:
  - GMS searches the Global Memory of various cluster nodes and finds Page X in Global Memory of node Q.
  - GMS copies Page X from Global Memory of node Q into Local Memory of node P.
  - This new copy of Page X in Local Memory of node P increases Local Memory size by 1. But since Local Memory + Global Memory = Physical Memory (DRAM), which is constant, Global Memory of node P needs to be reduced by 1.
  - So, GMS picks the oldest page in Global Memory of node P, say page Y, copies this page to Global Memory of node Q, and decreases the its Global Memory size by 1.
  - To summarize, Page Fault of Page X on node P increases Memory Pressure on node P.
    1. On node P:
       Local Memory size ++
       Global Memory size --
       Boundary of P moves down
    2. On node Q:
       Page Y traded for Page X.
       Boundary of Q remains unchanged.

Handling Page Faults – Case #2:

- What happens if a Page Fault happens on node P and it already has memory pressure (its Global Memory = 0)?
- Let's assume that Case #1 keeps happening repeatedly on node P.
  - The Local Memory will keep growing and the Global Memory will keep shrinking, until Global Memory becomes 0 and Local Memory takes over the whole Physical Memory (i.e. there is no community service on node P).
  - Case #2 is this common case of too much memory pressure on node P with 0 community service.
- Now, if a Page Fault happens on node P, then there is no option on node P except to throw out some page from the working set of its processes in order to make room for the new page.
- The page on node P that is chosen for replacement is called victim/replacement/evicted page. This page is usually the LRU (Least Recently Used) page.
- The boundary on both nodes P and Q remains unchanged.

Handling Page Faults – Case #3:

- What happens if the faulted page is not available in the clustered memory?
- When none of the peer nodes have the faulted page of node P, the only option is to get it from the local disk
- A Page Fault of a Page X in Local Memory of node P causes a fetch for this page from the local desk.
  - This increases Local Memory size by 1. Hence, decreasing Global Memory size by 1.
- GMS evicts any page Y from Global Memory of node P, copies it to a peer node R and shrinks node P's Global Memory size by 1.
  - Hence, on node P:
    Local Memory size ++
    Global Memory size --
    Boundary P moves down.
  - Note that node R is the node that has the globally oldest page in entire cluster.
- Now, on node R, there are 2 possibilities:
  - Page Y from Global Memory of node P is copied into Global Memory of node R:
    We know that Global Memory has only clean pages, hence the earlier data in the page being replaced is simply discarded. Thus, on node R, evicted Page Z discarded and replaced by Page Y. Boundary R remains unchanged.
  - Page Y from Global Memory of node P is copied into Local Memory of node R:
    NEEDS EXPLAINATION
    1. If Page Z is dirty: Page Z is swapped to disk and replaced by Page Y. Boundary R remains unchanged.
    2. If Page Z is clean: Page Z discarded and replaced by Page Y. Boundary R remains unchanged.

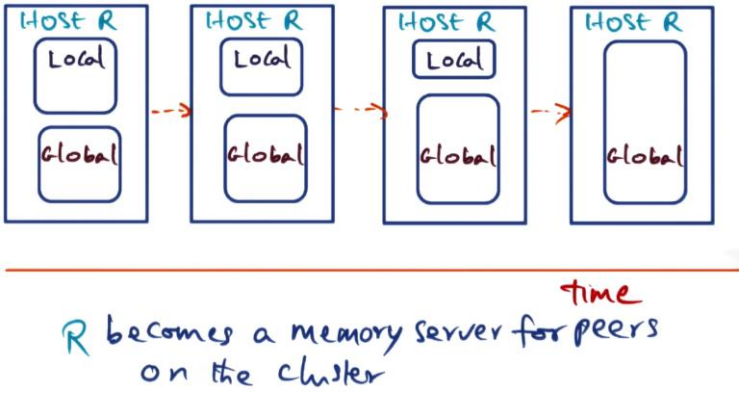
Handling Page Faults – Case #4:

- What happens if the faulted page of node P is a shared page?
- When a Page X is shared across nodes P and Q, but is currently present in the Working Set of a process on node Q. What happens if a process on node P page-faults the shared Page X?
- GMS searches for shared Page X in the peer cluster memory, finds it on node Q and copies it to node P.
- Increasing Local Memory size by 1 on node P causes a shrink of its Global Memory size by 1.
  - On node P:
    Local Memory size ++
    Global Memory size --
    Boundary P moves down
- However, since Page X is a shared page, GMS leaves the it in the Working Set of a process on node Q. Now, the same shared Page X is present in the Local Memory of both nodes P and Q. So, the total memory pressure in the cluster increases by 1 and eventually one page from one of the cluster nodes will need to be swapped out to disk.

- Any page from Global Memory of node P is chosen as the replacement/evicted page and copied to a node, say node R, that has the Globally Oldest Page.
  - Note: We choose an arbitrary page from Global Memory of node P and not the LRU page, because all pages in Global Memory are clean, peer pages and not used locally on node P.
- Now, on node R, there are 2 possibilities:
  - Page Y from Global Memory of node P is copied into Global Memory of node R:
    We know that Global Memory has only clean pages, hence the earlier data in the page being replaced is simply discarded. Thus, on node R, evicted Page Z discarded and replaced by Page Y. Boundary R remains unchanged.
  - Page Y from Global Memory of node P is copied into Local Memory of node R:
    NEEDS EXPLAINATION
    1. If Page Z is dirty: Page Z is swapped to disk and replaced by Page Y. Boundary R remains unchanged.
    2. If Page Z is clean: Page Z discarded and replaced by Page Y. Boundary R remains unchanged.

GMS with Idle Nodes:

- If a node remains idle for a long time-period, its Working Set is not utilized locally hence will be replaced by paged-out pages from peer cluster nodes.
- Eventually, a completely idle node becomes a memory server for peer cluster noes. Thus, the Local-Global Memory boundary is not static, but is dynamically changing in response to the Local Memory Pressure existing on that node.



Page Age Management:

- Geriatrics in GSM relates to Page Age Management:
  - Identifying the globally oldest page in the cluster.
  - Ensuring that the age management work is distributed well across all cluster nodes and does not burden a specific node.

- The Page Age Management is broken across the Space Axis and the Time Axis.
- Across the Time Axis, the page age management is broken into what is called Epochs
  - An Epoch is a "granularity" of page age management work done by a particular node.
  - The page age management work is either:
    1. Time-bound: The node does the page age management work for maximum time T duration.
    2. Space-bound: The node does the page age management work for maximum M replacements.
  - In other words, after max T duration or max M replacements, the current epoch is complete. The, a new node is picked as the new page age manager.
- The Page Age Management is distributed over multiple cluster nodes (Space Axis) and shifted over time onto different cluster nodes (Time Axis).
- T may be in the order of few seconds and M may be in the order of thousands of replacements.

How to Choose the Next Manager?

- The Manager for Page Age management is called an Initiator.
- A new Initiator is chosen at the start of every new Epoch.
- The older the page, the more likely to be unused and chosen for eviction.
- Every node sends the Page Age information for all Local and Global Memory Pages $\{L_p, \dots GP_n\}$ to the Initiator (Manager) node for the current Epoch.
- Out of all the pages in the cluster, the Initiator decides to do M replacements (evictions) of old pages in the cluster.
  - That is, if all the pages are sorted by Age, starting from newest to oldest page, then the list looks like:

$$Any\ Page\ Age < MinAge \ \rightarrow \ Active\ Page$$
$$Any\ Page\ Age \geq MinAge \ \rightarrow \ Replacement\ Page$$

  - The Initiator chooses the oldest M pages that are going to be replaced in current Epoch and determines $MinAge$ to be the minimum of the M replacements.
- Out of all the M replacement pages, the Initiator finds the percentage of pages for a particular node in those M replacement pages and calls this percentage the Node's Weight.

  Node Weight = Expected share of M replacements for a node.

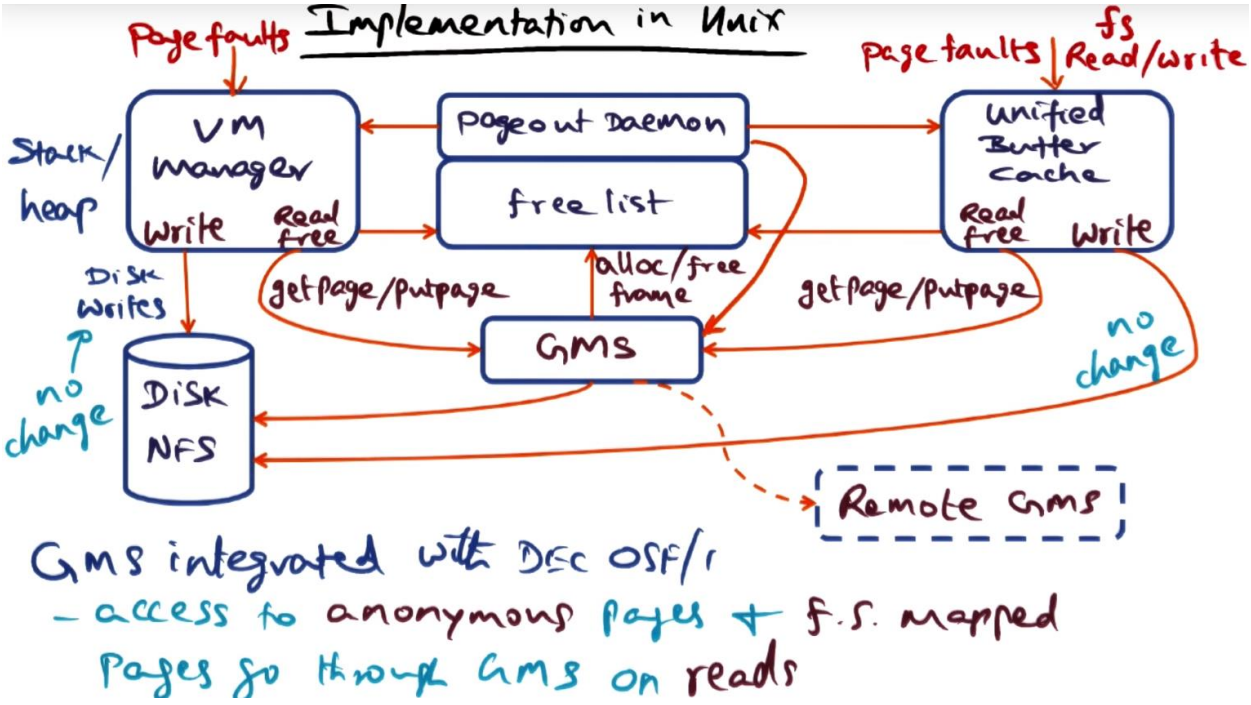
  The Initiator calculates the Weight for each node in the cluster.
- The Initiator sends to every node the following information:
  - $MinAge$ of M replacements
  - Weight Distribution of all cluster nodes: Weight of each node W_i for all cluster nodes V_i.
- Each node receives $\{MinAge$, Weight Distribution$\}$ from the Initiator. The node that has the Highest Weight in Weight Distribution is the Least Active or Most Inactive. Hence, each node locally chooses the node with the highest Weight in the Weight Distribution as the Initiator for the next Epoch. This decision is made locally without any coordination with any other cluster node.

- The Initiator uses the principle of "using the past to predict the future". That is, use the page age information (past) to predict where the replacements will happen (future).
- When a page fault happens on a node P and a new page X is brought into Local Memory of node P, an old page Y from Global Memory of node P is chosen to be evicted/replaced.
- Node P locally chooses the following approach to decide which page Y needs to be evicted:
  - If $Page\ Y's\ Age \geq MinAge$, then this page is a Replacement Page to be discarded in the next Epoch.
  - If $Page\ Y's\ Age < MinAge$, then this page is an Active Page and so send it to a peer cluster node with the highest Weight among the Weight Distribution Vector.
  - Thus the Page Age Management (Geriatric Management) is approximating a "Global LRU" by making an approximate estimate of what is going to happen in future.
  - Note that each node makes decisions locally without any coordination with any other cluster node.
- In summary:
  - At the beginning of each epoch, GMS Initiator globally computes the following to get all Page Age info:
    1. $MinAge$ of M replacements.
    2. Weight Distribution of all cluster nodes.
  - GMS then sends these information to all cluster nodes.
  - Using these information, each cluster node then makes local decisions in terms of what to do with a page that is chosen as an eviction candidate:
    1. Discard the page if it will be replaced soon.
    2. Send it to peer cluster memory if it is active.

GSM Implementation:

- The basic idea of GMS is that instead of using disk as a paging device, use the cluster memory.
- The GMS authors used DEC's OSF/1 operating system to implement GMS.
- There are two components of the OSF/1 OS's Memory system:
  - Virtual Memory system: This is devoted to managing the page faults that occur for the process address space, in particular, the stack and the heap, and to get these missing pages from disk. These pages are called Anonymous Pages, because a Virtual Page is housed in a Physical Page Frame and when a page is replaced, that same Physical Page Frame is used to host a new Virtual Page.
  - Unified Buffer Cache: This is the File System Cache used to cache pages from frequently used files. The Unified Buffer Cache is responsible for handling page faults to memory-mapped files as well as for handling explicit read and write calls that an application makes to file system.

- In a typical OS:
  - The writes from the Virtual Memory Manager and Unified Buffer Cache go to the disk.
  - A page fault causes read of the required page from disk.
  - When a physical page frame is freed, it is thrown back into the free list.
  - The page out daemon periodically discards clean pages and swaps-out dirty pages to disk in order to avoid this expensive activity of writing to disk during a future page fault.
- After GMS is integrated into the OS:
  - The writes from the Virtual Memory Manager and Unified Buffer Cache go to the disk.
  - A page fault causes read of the required page from GMS instead of reading it from disk.
  - When a physical page frame is freed, it is thrown back into the free list.
  - The page out daemon periodically gives clean pages to GMS for later retrieval from peer node and swaps-out dirty pages to disk.
- The tricky part is collecting Page Age information required for the Global LRU approximation:
  - Unified Buffer Cache (UBC) calls are intercepted by modifying UBC to collect Page Age information for pages that are housed in the UBC.
  - Changing the Virtual Memory Manager is complicated because the memory access performed by a process happens in hardware on the CPU and there is no easy way to intercept it. The OS does not see the individual memory access that a user program is making. So, the trick used is to have a daemon periodically dump TLB contents into a GMS structure and use this information to derive Age information for all Anonymous Pages that are being handled by the Virtual Memory.



GSM Data Structures:

- The basic idea of GMS is that instead of using disk as a paging device, use the cluster memory.
- Virtual Address (VA) is converted to Universal Identifier (UID) to be used cluster-wide using information from the Virtual Memory System and the Unified Buffer Cache like:
  - IP-address of the node containing the Virtual Address.
  - Disk Partition that contains a copy of the page that corresponds to the Virtual Address.
  - i-node data structure of the file that corresponds to the Virtual Address.
  - Offset in that file for the page that corresponds to the Virtual Address

  Note that UID space spans the entire cluster.
- Page Frame Directory – PFD: This is like a Page Table that converts the Universal Identifier (UID) to a Physical Page Frame Number (PFN) hosting that Virtual Address
- Global Cache Directory – GCD:
  - This is a partitioned, cluster-wide Hash Table, used to distribute management of mapping from UID to node hosting corresponding PFD so as to avoid the static mapping problem of over-burdening any single node.
  - Given a UID, GCD will tell us which node has the PFD corresponding to this UID.
- Page Ownership Directory – POD:
  - Given a UID, POD says which node has the corresponding GCD.
  - The POD is replicated on all the cluster nodes and has up-to-date information on each node.
  - The UID space that spans the entire cluster is partitioned into sets of ownership regions called Page Ownership and every node is responsible for a portion of the UID space, present in the POD for that node.
  - If the nodes in a cluster remain the same, then POD is static, but if nodes are added or deleted, then the POD needs to be redistributed, which is usually rare.

GSM and Page Faults:

- Whenever a Page Fault happens, we go through these steps:
  - The node converts the VA to a UID.
  - Check the owner of the UID using the POD.
  - Sends the UID to its owner.
  - The owner node checks its GCD to determine the node the contains the PFD for this UID.
  - The UID is then sent to the node that has the PFD, which in turn converts the UID to a PFN and retrieves the page. Then it sends the page to the starting node.
- The common case is for a page to be non-shared (i.e. based on a request of a local process on a node), and POD and GCD are on the same node and hence GMS can directly go to the PFD node to get the required page. So, the page fault service is quick in most cases since it uses only one remote network communication..

- It's possible that a node may not have the desired UID-PFN mapping. This scenario is quite rare:
  - After reaching the node expected to have the PFD of interest, GMS finds that the PFD does not have the desired UID-PFN mapping.
  - This could happen when the node containing this PFD evicts the corresponding page and sends it to a peer cluster node and the other distributed data structures are still being updated.
  - Another scenario is when the POD information is stale due to new additions or new deletions of nodes in the cluster and the distributed data structures are still being updated.
  - In such scenarios, GMS does a re-lookup of the POD data structure, hoping that the POD data structure may have been updated correctly by this time and the subsequent lookups will help GMS get routed to the correct node having appropriate PFD for the UID-PFN mapping.
  - This is expected to happen very rarely as compared to the common case described earlier.