## Today

Distributed Subsystems (Lesson 7)

⇒ * Global memory system

⇒ * Distributed shared Memory

## Wednesday

* Distributed file systems

# Question

Fill in the table with what happens to the boundary between Local and Global on each node

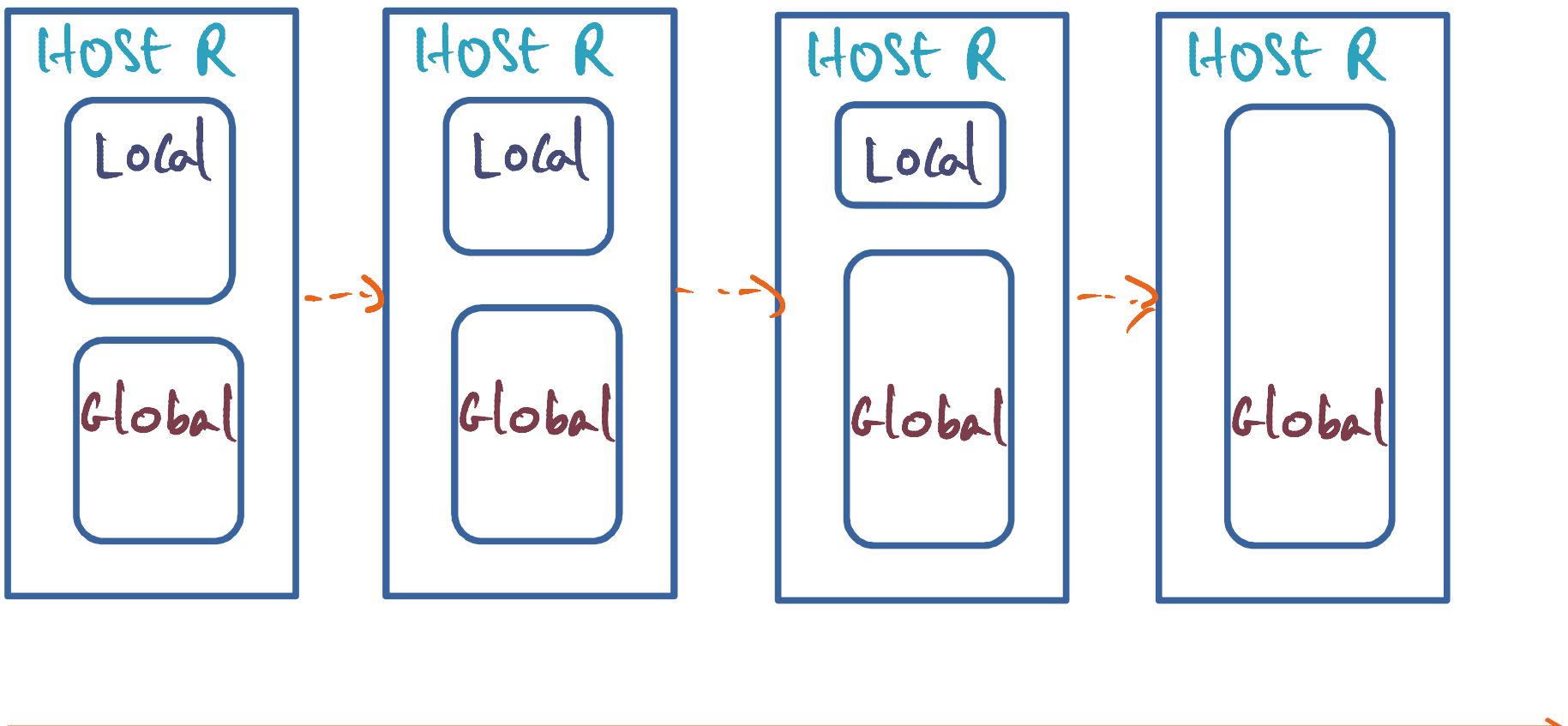
| Faulting Page X | Faulting node P | | Node Q with page X | | Node R with LRU page | |
|---|---|---|---|---|---|---|
| | L | G | L | G | L | G |
| in Q's global | | | | | | |
| in Q's global P's global empty | | | | | | |
| on disk | | | | | | |
| actively shared with Q | | | | | | |

# Question

Fill in the table with what happens to the boundary between Local and Global on each node

| Faulting Page X | Faulting node P | | Node Q with page X | | Node R with LRU page | |
|---|---|---|---|---|---|---|
| | L | G | L | G | L | G |
| in Q's global | +1 | −1 | no change | | no change | |
| in Q's global P's global empty | no change | | no change | | no change | |
| on disk | +1 | −1 | Not applicable | | −1 | +1 |
| actively shared with Q | +1 | −1 | no change | | −1 | +1 |

# Behavior of Algorithm



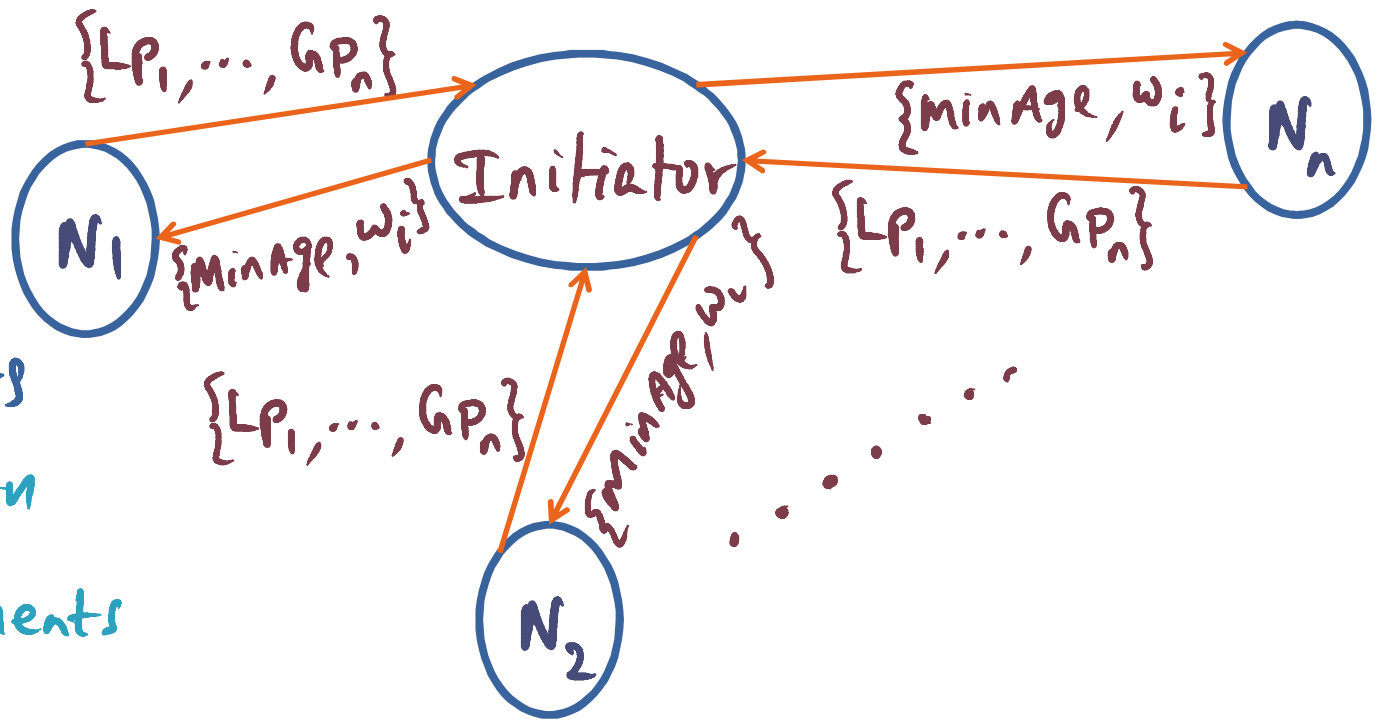

R becomes a memory server for peers on the cluster

# Geriatrics!!

Epoch parameters

- T max duration
- M max replacements

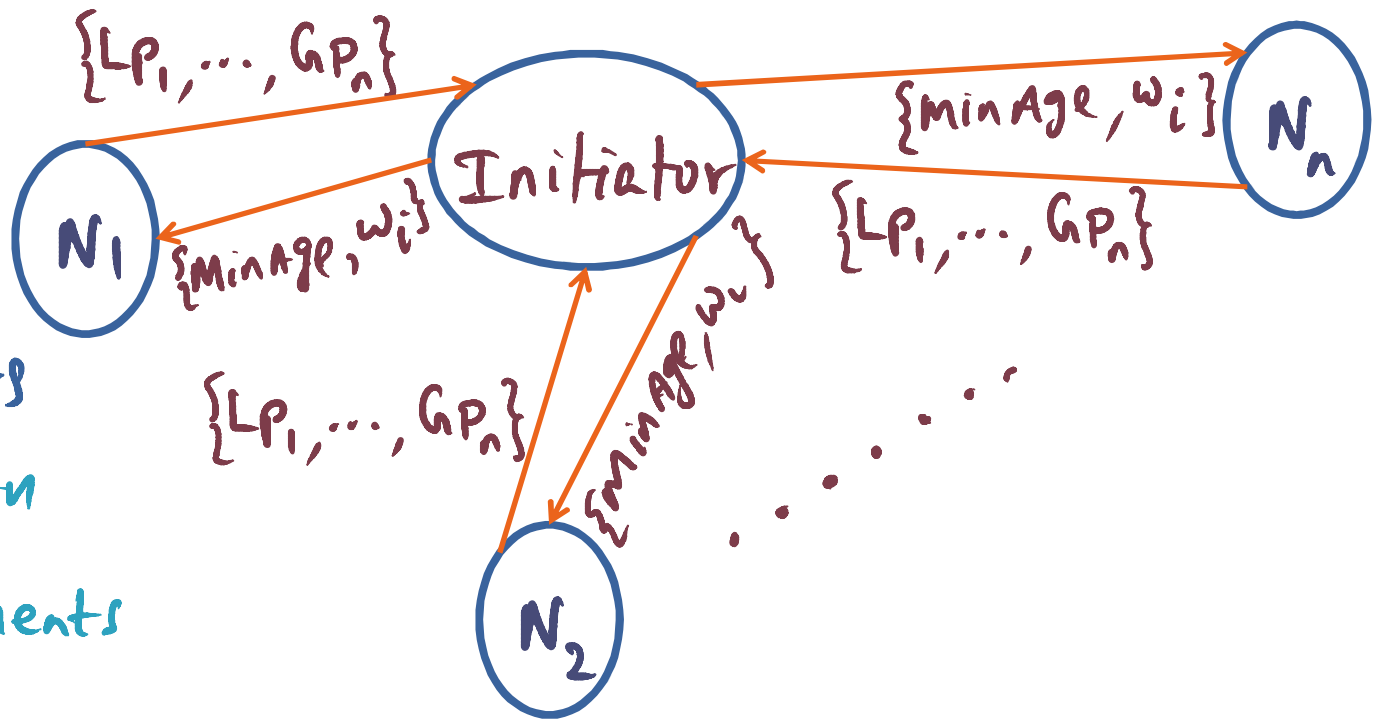
# Geriatrics!!




Epoch parameters
- T max duration
- M max replacements

Each Epoch
- Send age info to initiator
- receive {minAge, wi} ∀ i

# Geriatrics!!

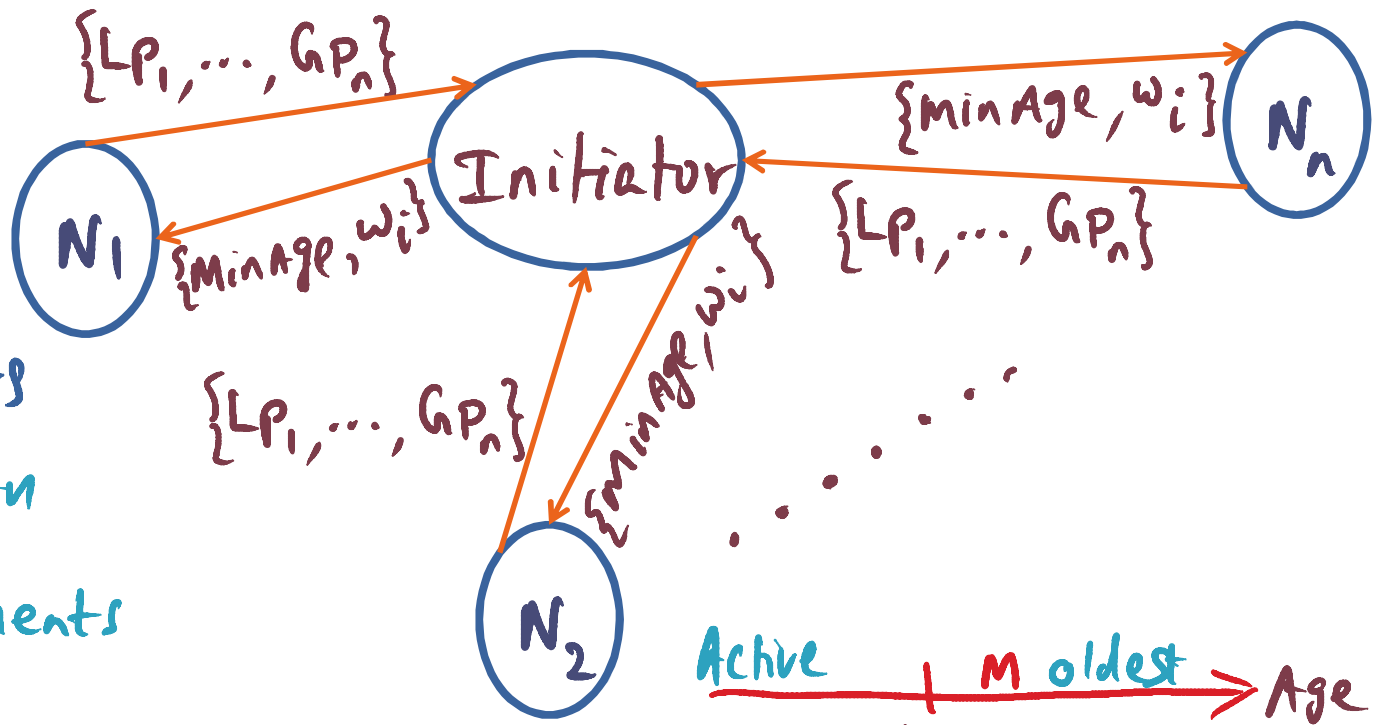



Epoch parameters
- T max duration
- M max replacements

Each Epoch
- Send age info to initiator
- receive {minAge, wi} ∀ i

Initiator for next epoch
- node with Max(wi)

# Geriatrics!!

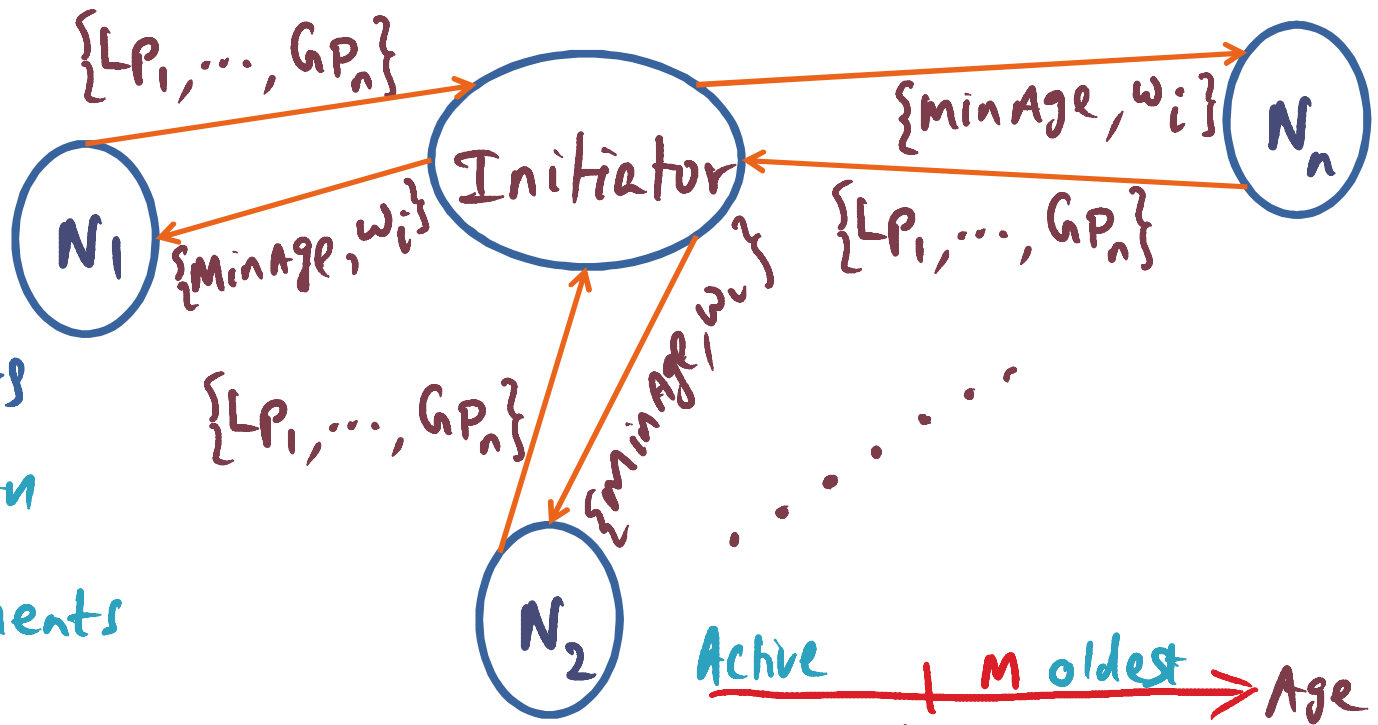



Epoch parameters
- T max duration
- M max replacements

Each Epoch
- Send age info to initiator
- receive {minAge, wi} ∀ i

Initiator for next epoch
- node with Max(wi)

# Geriatrics!!




Epoch parameters
- T max duration
- M max replacements

Each Epoch
- Send age info to initiator
- receive {minAge, wi} ∀ i

Initiator for next epoch
- node with Max(wi)

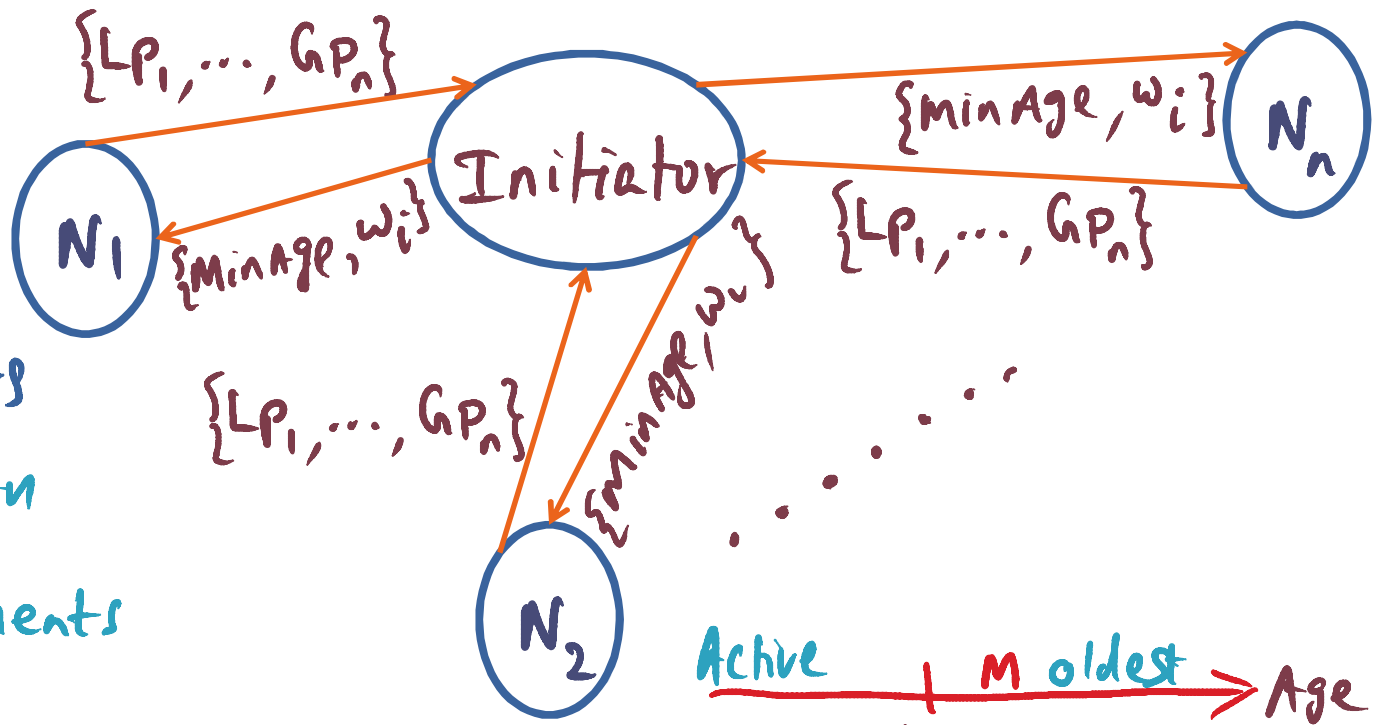
Action at a node on page fault
Page y eviction candidate

Age(page y) > MinAge ⇒ discard

Age(page y) < MinAge
⇒ Send to peer $N_i$

Think Global
Act local !!

# Geriatrics !!




Epoch parameters
- T max duration
- M max replacements

Each Epoch
- Send age info to initiator
- receive {minAge, wi} ∀ i

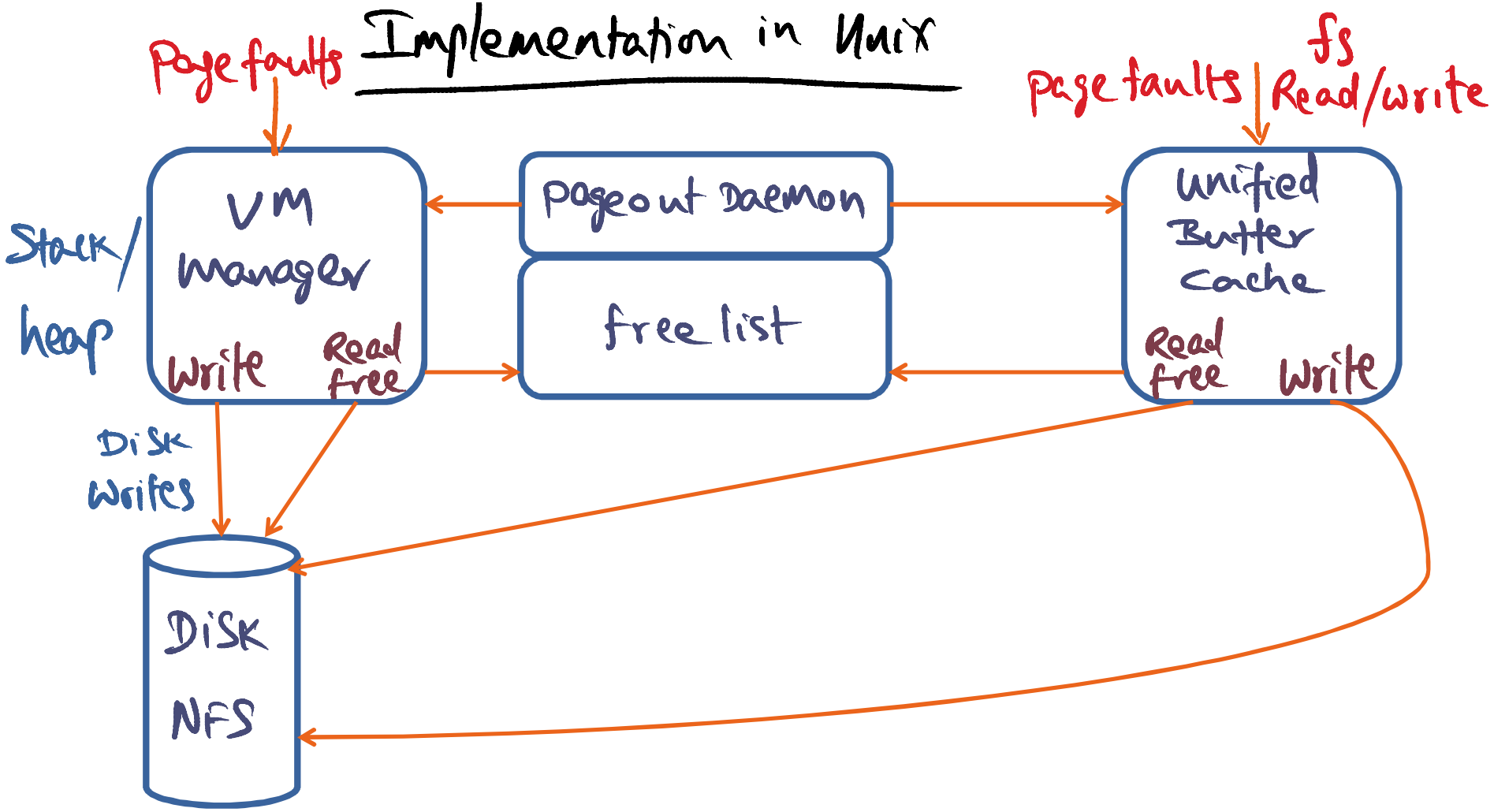
Initiator for next epoch
- node with max(wi)

Action at a node on page fault
Page y eviction candidate

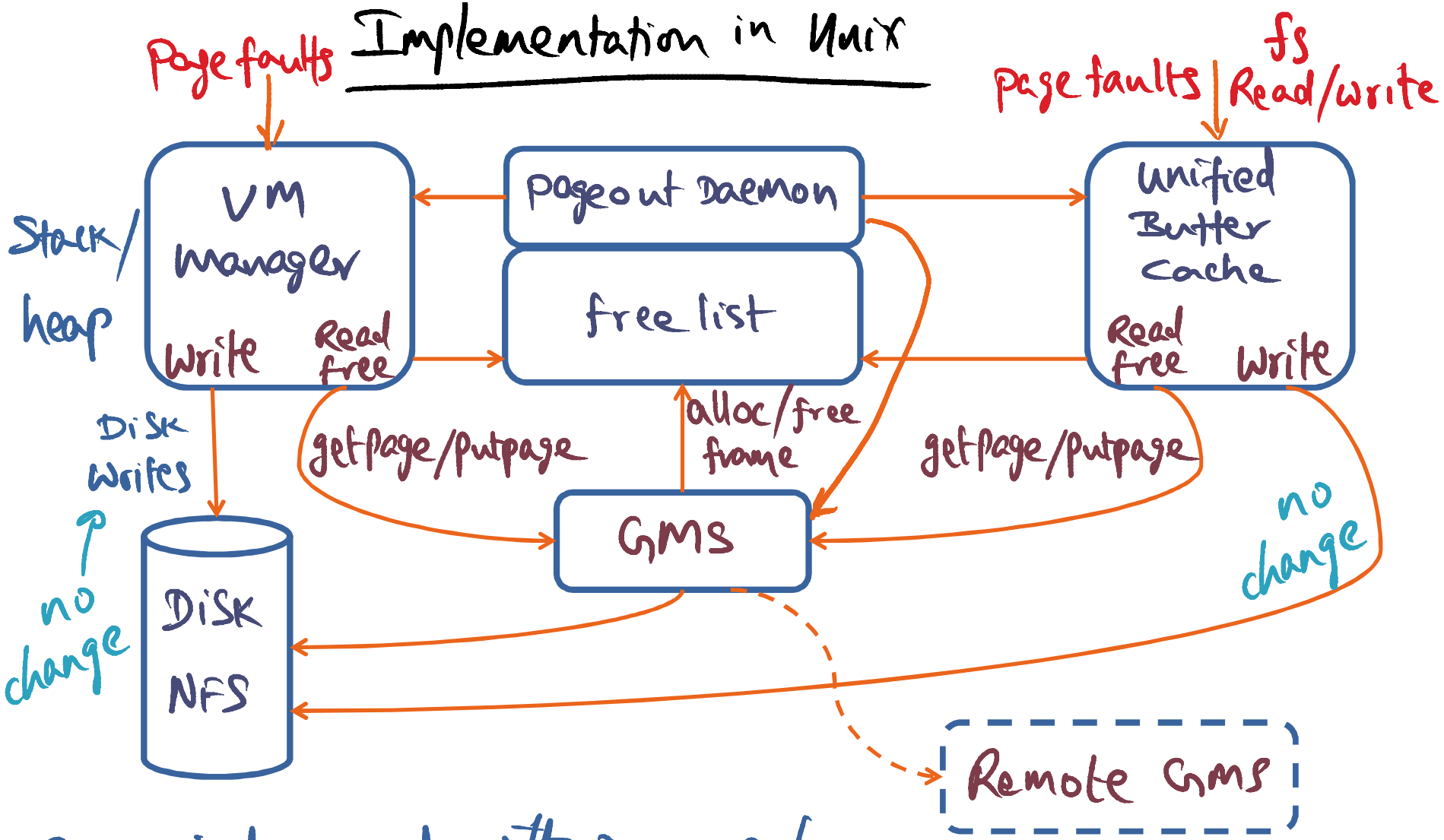
Age(page y) > MinAge ⇒ discard

Age(page y) < MinAge
⇒ send to peer $N_i$

# Implementation in Unix



# Implementation in Unix




GMS integrated with DEC OSF/1

- access to anonymous pages + f.s. mapped

Pages go through GMS on reads

# Data structures

VA → UID


- derived from VM + UBC

Data structure for

UID → physical page frame ?

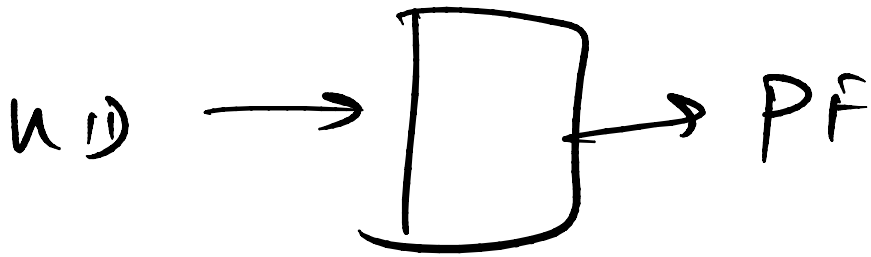
# Data structures

VA → UID


- derived from VM + UBC

Data structure for

UID → physical page frame ?



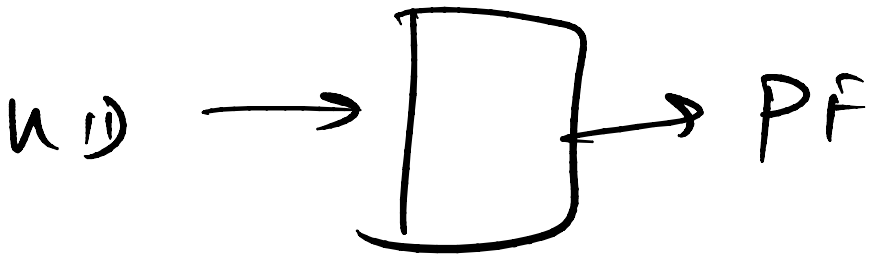
# Data structures

VA → UID


- derived from VM + UBC

Data structure for

UID → physical page frame ?




N x P x V

nodes processes VP per process

# Data structures

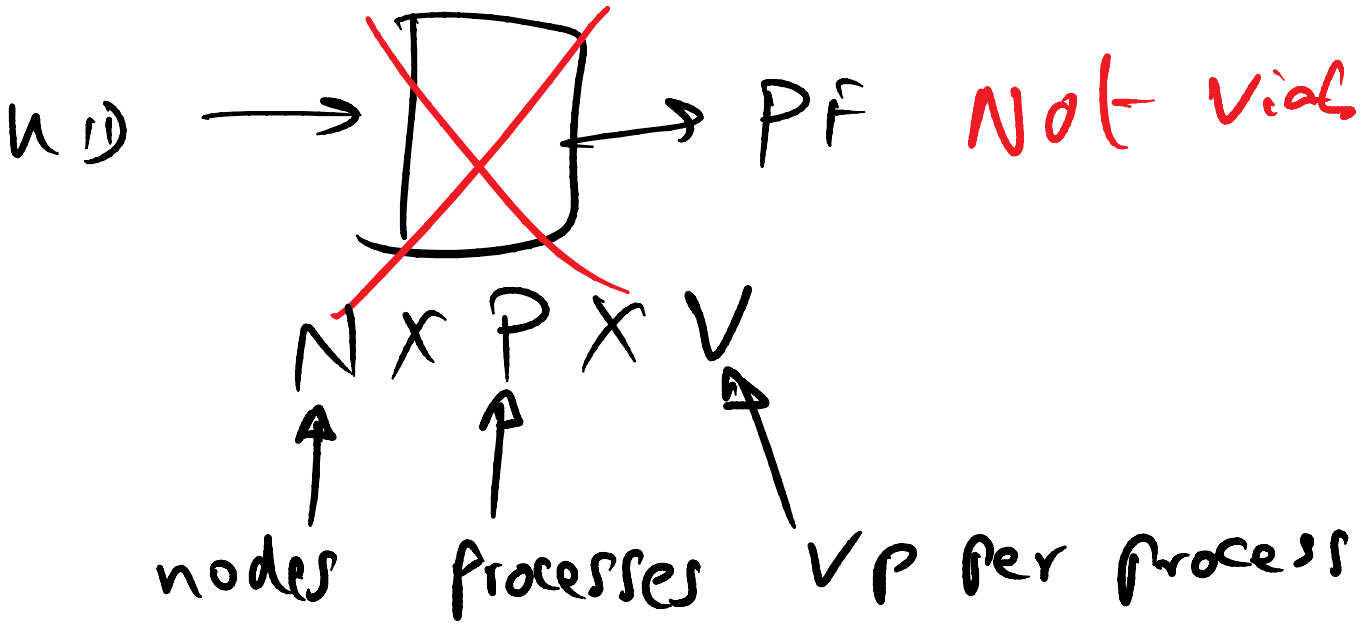
VA → UID

| IP_Addr | disk partition | i-node | offset |
|---|---|---|---|

- derived from VM + UBC

Data structure for

UID → physical page frame ?



# Twin Goals

* reduce communication
* scalable wit number of nodes
  ⇒ don't burden any one node

# Data structures

VA → UID


- derived from VM + UBC

Data structure for

UID → physical page frame ?

# Data structures

VA → UID


- derived from VM + UBC

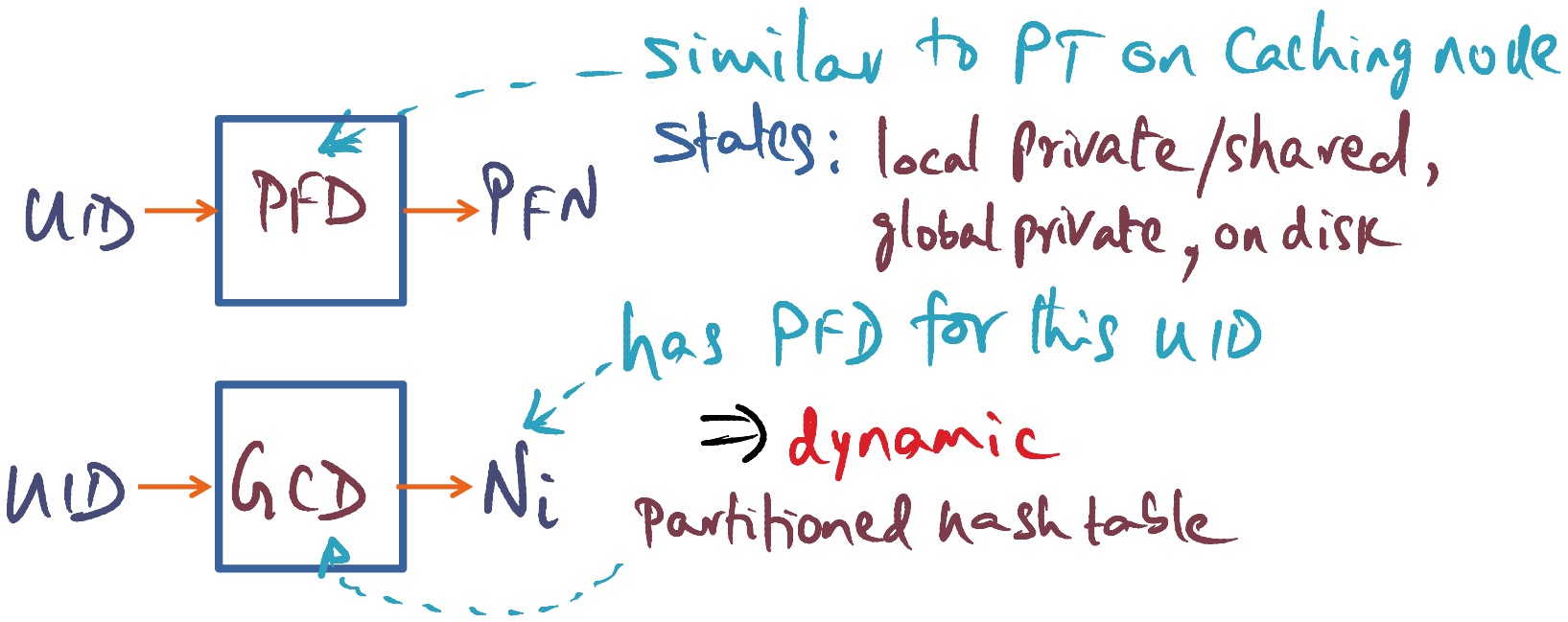

- - - similar to PT on caching node

States: local private/shared, global private, on disk

# Data structures

VA → UID

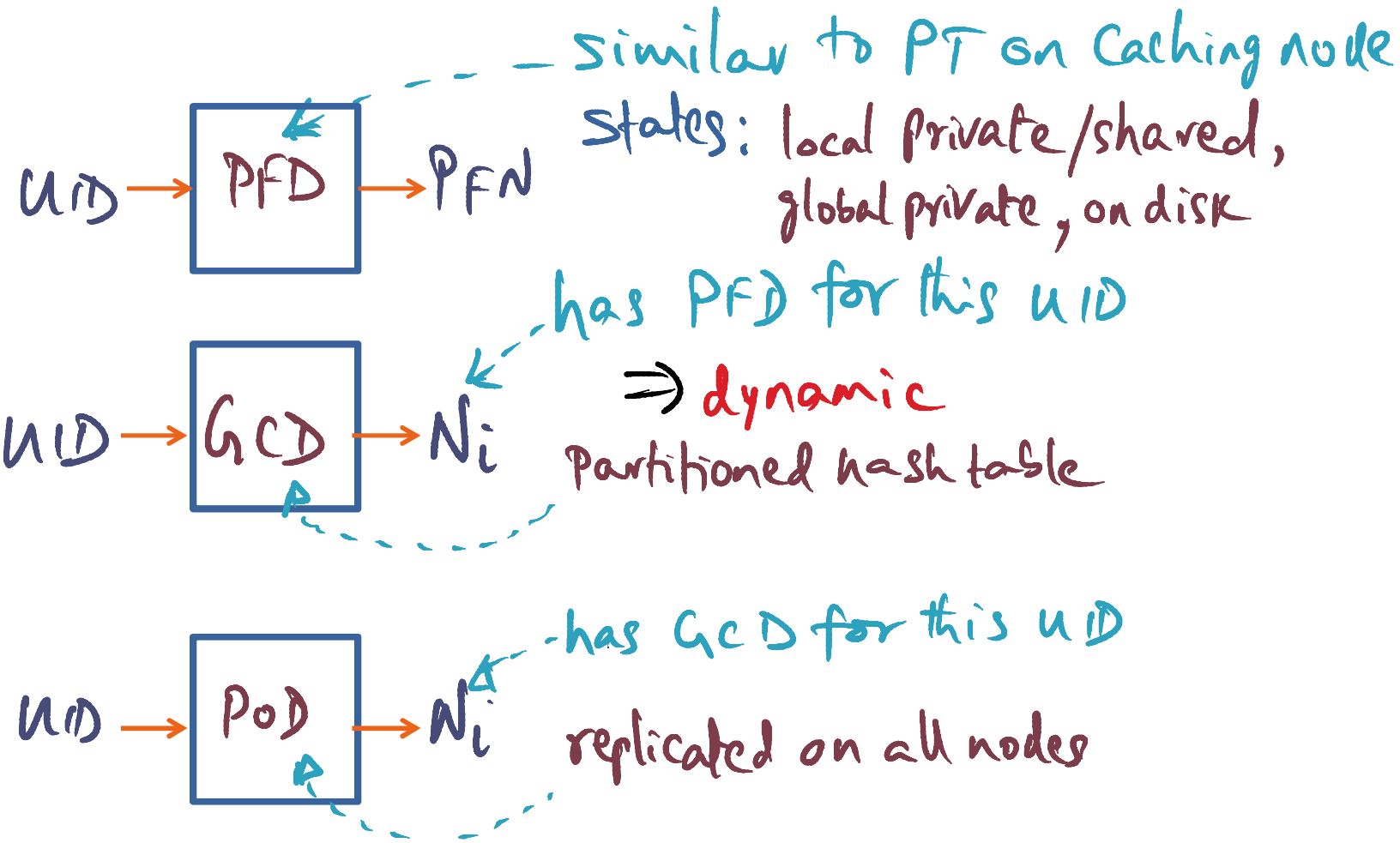

- derived from VM + UBC



# Data structures

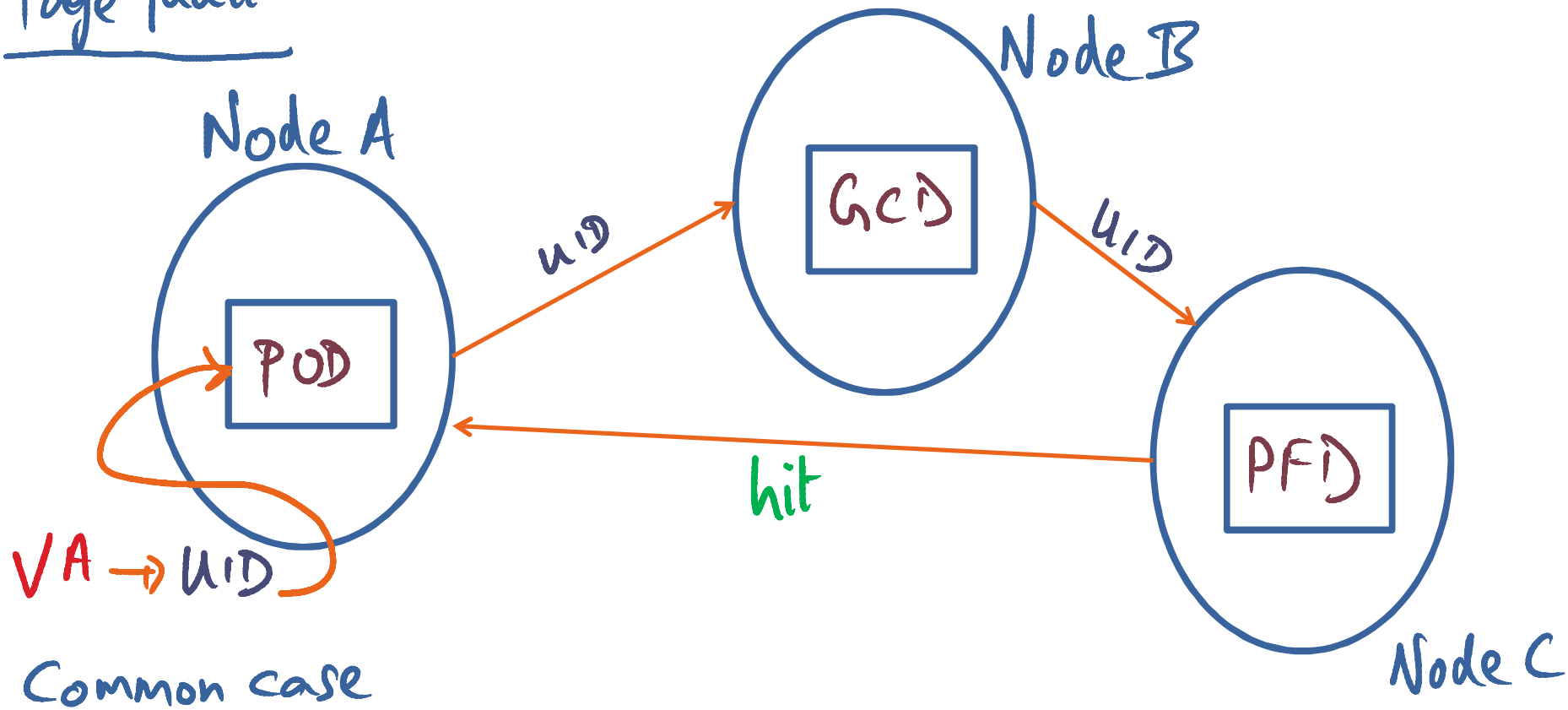
VA → UID


- derived from VM + UBC



# Putting the Data structures to work

## Page fault

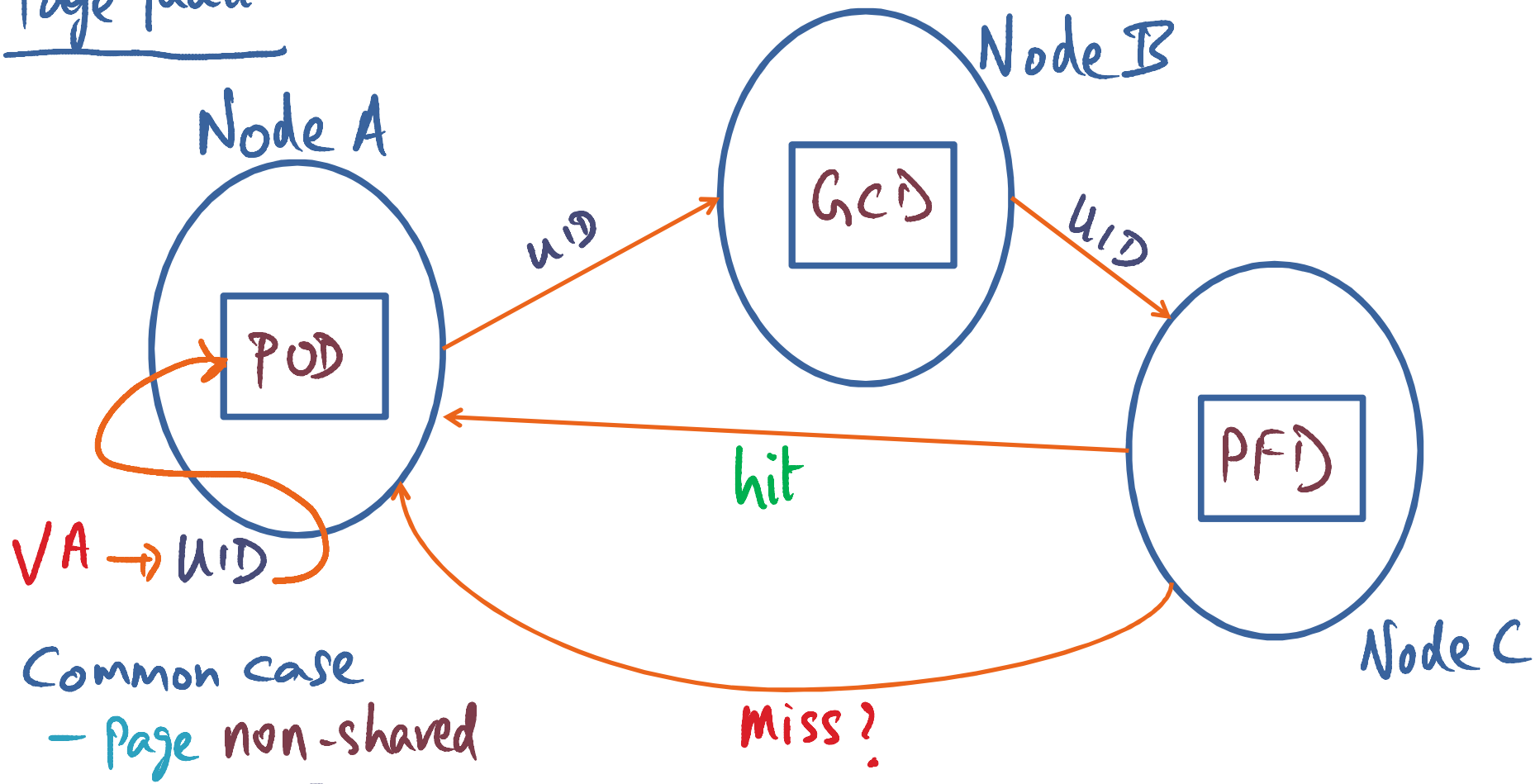



Common case

- Page non-shared
  ⇒ A + B same
- Page fault service quick

# Putting the Data structures to work

## Page fault

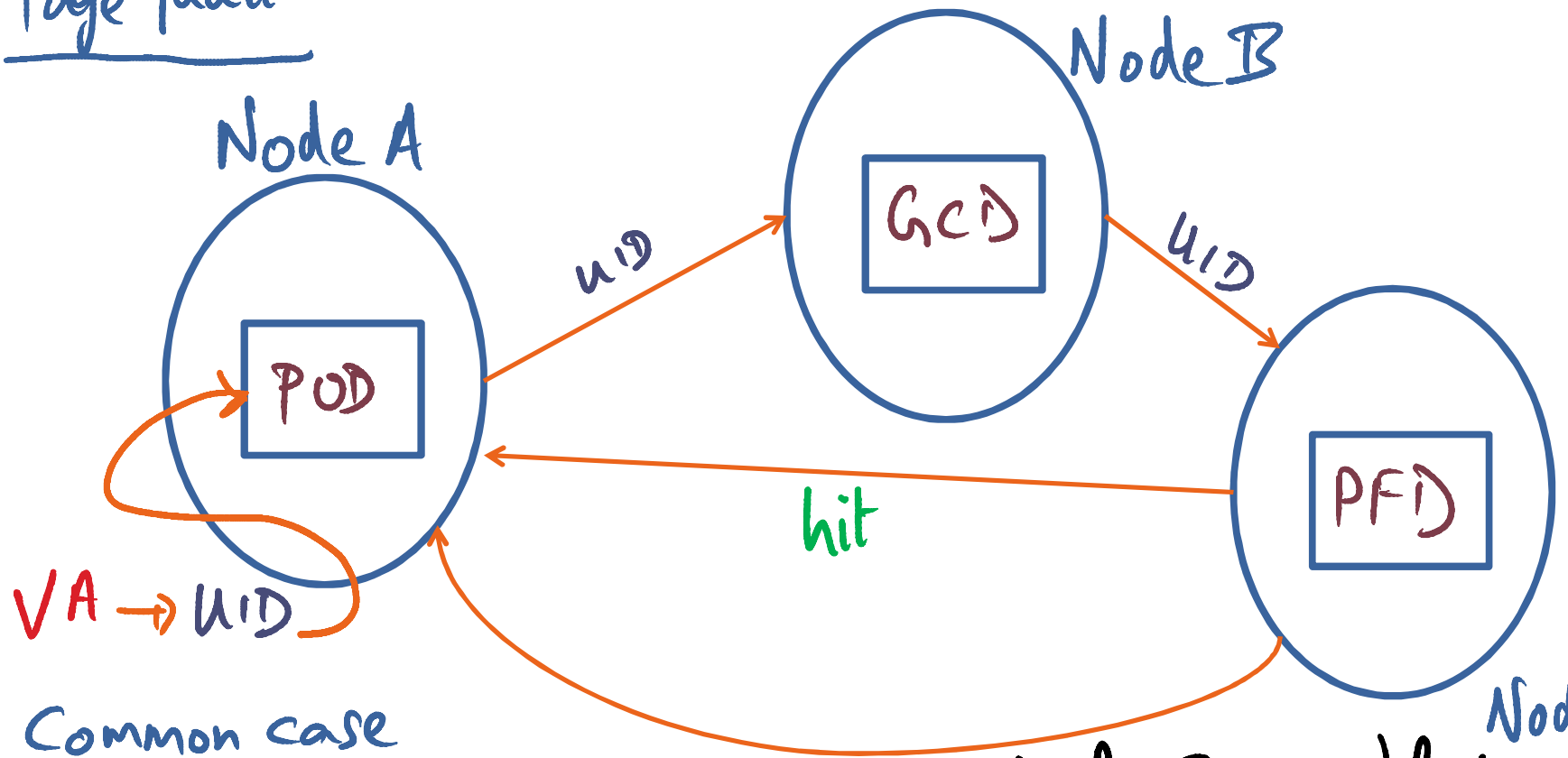



Common case

- Page non-shared
  ⇒ A + B same
- Page fault service quick

# Putting the Data structures to work

## Page fault

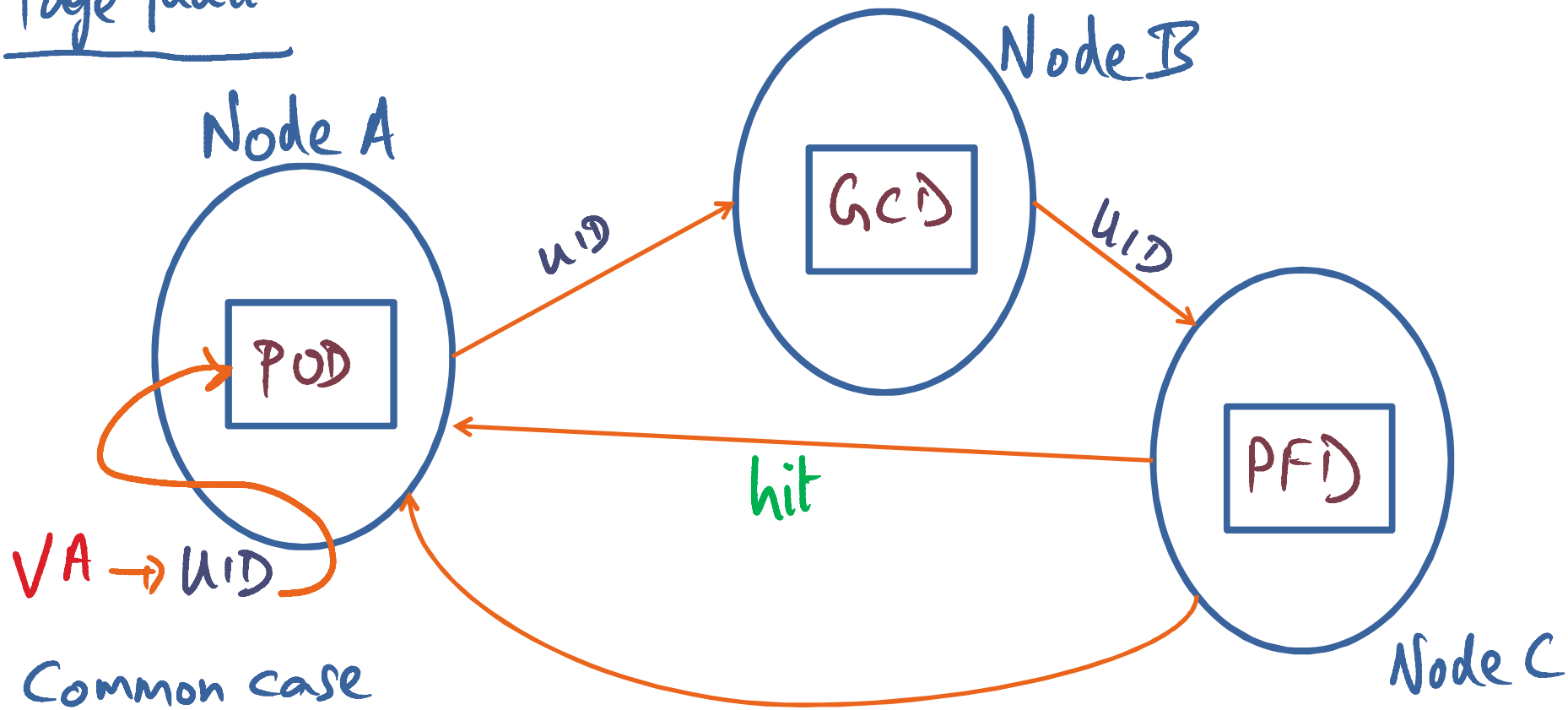



Common case
- Page non-shared
  ⇒ A + B same
- Page fault service quick

Miss? Node C could have evicted the page, and the GCD update is in progress

# Putting the Data structures to work

## Page fault

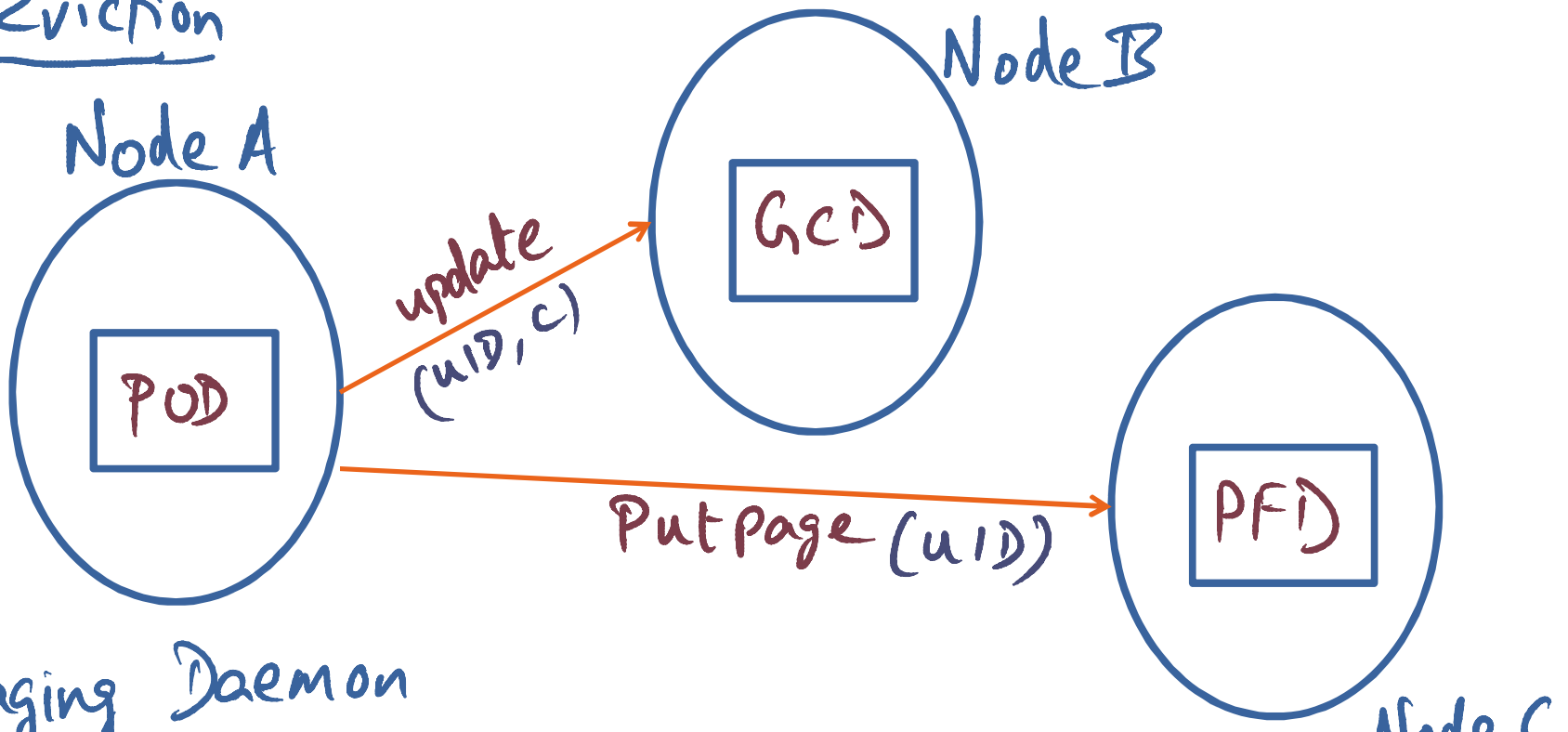



Common case
- Page non-shared
  ⇒ A + B same
- Page fault service quick

⇒ Uncommon Case
- POD changing due to node addition/deletion

# Putting the Data structures to work

## Page Eviction




Paging Daemon

- Freelist below threshold
- putpage oldest pages
- update GCD, PFD for the UIDs

# Question

Fill in the table with what happens to the boundary between Local and Global on each node

| Faulting Page X | Faulting node P | | Node Q with page X | | Node R with LRU page | |
|---|---|---|---|---|---|---|
| | L | G | L | G | L | G |
| in Q's global | | | | | | |
| in Q's global P's global empty | | | | | | |
| on disk | | | | | | |
| actively shared with Q | | | | | | |

# Question

Fill in the table with what happens to the boundary between Local and Global on each node

| Faulting Page X | Faulting node P | | Node Q with page X | | Node R with LRU page | |
|---|---|---|---|---|---|---|
| | L | G | L | G | L | G |
| in Q's global | +1 | −1 | no change | | no change | |
| in Q's global P's global empty | no change | | no change | | no change | |
| on disk | +1 | −1 | Not applicable | | −1 | +1 |
| actively shared with Q | +1 | −1 | no change | | −1 | +1 |

Write down key takeaways and give to your neighbor

# Key takeaways

- Heavy lifting to be done to take a concept to implementation.
  - non-trivial intellectual exercise in building distributed subsystems
- What is enduring in a research exercise like this one?
  - The concept of paging across the network is an interesting thought experiment but it may not be feasible exactly for the environment in which the authors carried out the research (workstation clusters connected to a LAN). Each workstation in that setting is a private resource of an individual who may or may not want his resources (memory in particular) shared with others…
  - On the other hand, data centers are powered by large-scale clusters (> 1000s of processors in a LAN). No node is individually "owned" by a user. Could this idea of paging across the LAN be feasible in that setting? Perhaps.
- Even beyond the thought experiment itself, what is perhaps more enduring are the techniques (distributed data structures and algorithms) for taking the concept to implementation