Page Migration / Replication

- Ratio between Remote & Local mem access
  - → CC-NUMA: 3-5-10 times
  - → CC-NOW: 10-20 times
- Data locality is potentially the most important issue in performance of NUMA
  - → compounded in time-shared servers
- What to do? page mig: follow the process
  page repl: for read-shared pgs

Results of Verghese et al

- OS-induced pg mig/repl: → +29% perf. over a first-touch policy
- Both pg mig, repl are required
- Not enough to base mig/rep on TLB misses
- Need to modify kernels so that overheads do not negate the perf improvement

Related Work

- Mig/Repl in machines that support shmem in SW → Mig/Repl is required for correctness/not opt.
- Systems w/o coherent caches: Use TLB miss information + page freezing/defrosting
- CC-NUMA: if locality, caches capture remote data and, therefore, pg mig/rep unnecess.

Pbm Statement

- Minimize runtime pgm by reducing component of mem stall → convert remote to local accesses
  → not suffer too much overhead in OS
- Find pages suffer most of remote misses
- Access patterns to a page
  1. Primarily by single process : possibly migrate them when process migrates,
     → code of sequential app
     → data " " "
     → disjoint data of parallel app

- Pages accessed by multiple processes (mostly RD only)
  - → candidates for replication
    - code of parallel apps
    - code of concurrent seq apps
    - read-mostly data of parallel app
- Pgs accessed RD-WR by multiple processes
  - → not candidates

Cost of Pg Mig/Repl

- Gathering info: → count all cache misses or
  → " TLB "
  → time sampling of each/either
- Kernel overhead:
  → allocate new page
  → change physical mappings
  → flushing TLBs
  → eliminating replicas of replicated pg
- Data movement
- Increase mem use

Decision Tree Based on Cache Misses

- Decision may be triggered on any miss
- Only interesting pages: if high misses

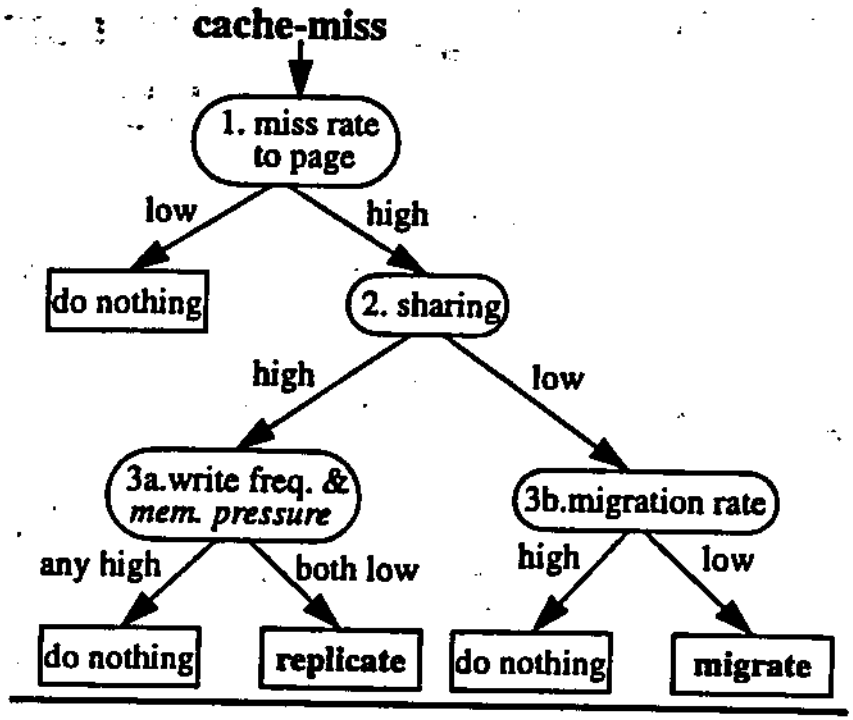


FIGURE 1: Replication/Migration decision tree.
The flowchart shows the decision process for a page to which a cache miss is taken. The possibilities are to replicate, migrate, or do nothing.

Kernel Mechanisms

- Modified IRIX 5.2 (bus-based OS)
- To track rates of misses: counters w/ periodic reset
- For each page: miss cntr per processor
  migrate cntr
  write cntr
- Only consider "hot pages" that are in a remote location

Procedure

- If cntr > trigger threshold ⇒ hot page
- if remote:

  if miss cntr on any other processor > sharing threshold

  ⇓

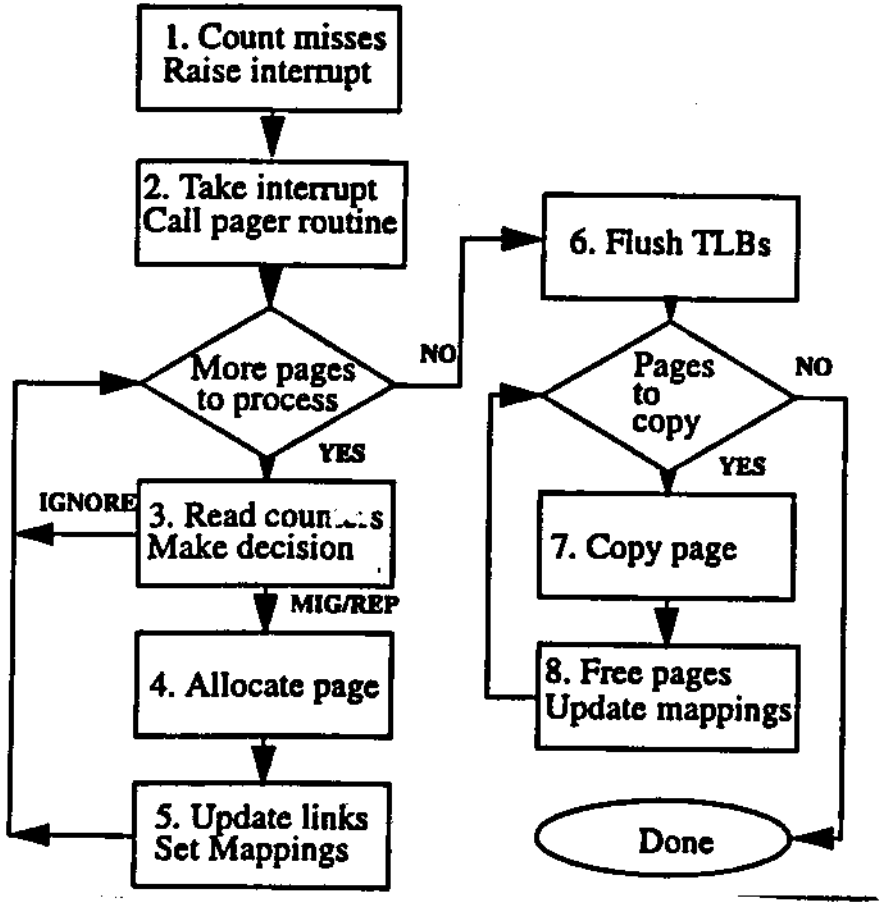
  candide for repl
  otherwise candide for mig

  if cand repl and write cntr < write threshold ⇒ repl

  if cand mig and mig cntr < mig thresh ⇒ mig

Implementation

- Dir controller keeps miss counters for local pgs
  generates intr when pg crosses trigger
  interrupt
- Low priority intr handler
- Directory controller attempts to collect multiple pages before generating an intr
- To handle writes to replicated pages:
  - → pg table entries for replicated pgs → marked RD only
  - → if write to pg → fault handler that collapses replicas



Parameter	Semantics
Reset Interval	Number of clock cycles after which all counters are reset.
Trigger Threshold	Number of misses after which page is considered hot and a migration/replication decision is triggered.
Sharing Threshold	Number of misses from another processor, making a page a candidate for replication instead of migration.
Write Threshold	Number of writes after which a page is not considered for replication
Migrate Threshold	Number of migrates after which a page is not considered for migration.

TABLE 1: Key parameters used by the policy. These parameters are used along with the counters to approximate rates. The counters used by the policy include a per-page per-processor miss counter, a per-page write counter, and a per-page migrate counter.

Changes to IRIX

- Replication support: link their page descriptors together
- Finer grain locking: (for VM-related structs)
- Pg table back mappings: inverted pg table (links) to facilitate mapping changes

Evaluation Methodogy

- Simulations (includes OS + Appl)
- 8 processor NUMA / NOW
- 64-entry TLB
- ratio: NUMA : 4 NOW : 10
- parameters: Trigger treshold 128 ...

compared to

first-touch

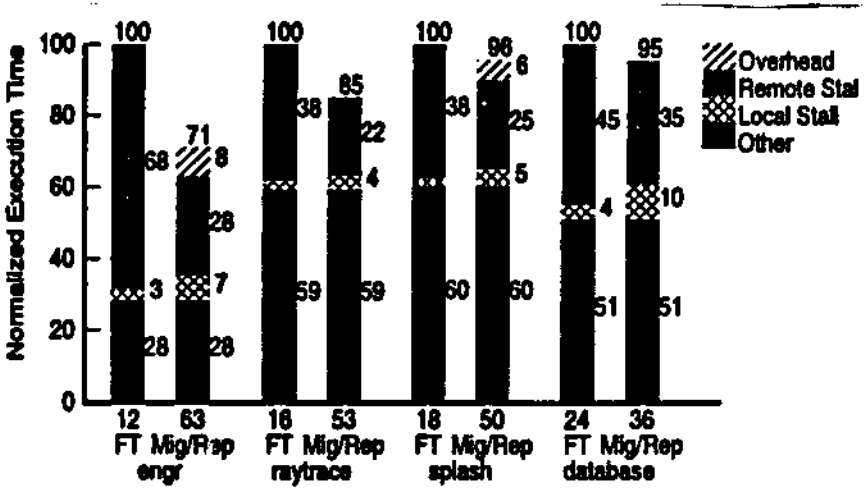


FIGURE 3: Performance improvement of the base policy (Mig/Rep) over first touch (FT). The execution time is divided into the kernel overhead for migration and replication, the remote and local stall times, and all other time. The figure shows the percentage of misses to local memory at the bottom of each bar.

Name	Contents	Notes
Engr.	6 Flashlite 6 Verilog	multiprogrammed, compute-intensive serial applications. Affinity sched
Raytrace	Raytrace	parallel graphics application (rendering a scene). Locked
Splash	Raytrace Volrendering Ocean	multiprogrammed, compute-intensive parallel applications. Space part.
Database	Sybase	commercial database (decision support queries). Locked
Pmake	4 four-way parallel Makes	software development (compilation of gnuchess)

TABLE 2: Description of the workloads. For each workload, the table lists the applications in the workload and a short description of the workload. All the workloads are run on an eight processor configuration, except the database workload that uses four.

Work load	Hot Pages	% Migrate	% Replicate	% No Action	% No Page
Engr.	7,728	55	27	12	6
Ray.	2,934	34	31	35	0
Splash	6,328	36	22	18	24
DB	2,003	13	2	85	0

TABLE 4: Breakdown of actions taken on hot pages. For each workload the hot pages are broken down into the percentage of migrations, replications, instances no action was taken, and failures because no physical page was available on the local node.

Baseline Performance

- Performance gains depend on:
  - → contribution of user stall time
  - → fraction misses local
  - → Kernel ouhead
- Both mig / repl are necessary
- Sometimes: no pg available for action
  (mem in some nodes is exhausted)
- Modest gains

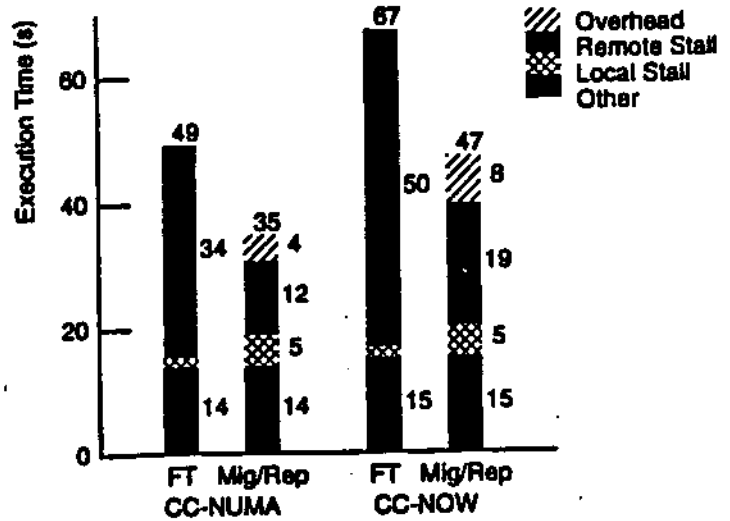


FIGURE 5: Performance comparison of the CC-NUMA and CC-NOW configurations for the engineering workload. For each of the two configurations, the non-idle execution time is shown for the first touch (FT) and base (Mig/Rep) policies.

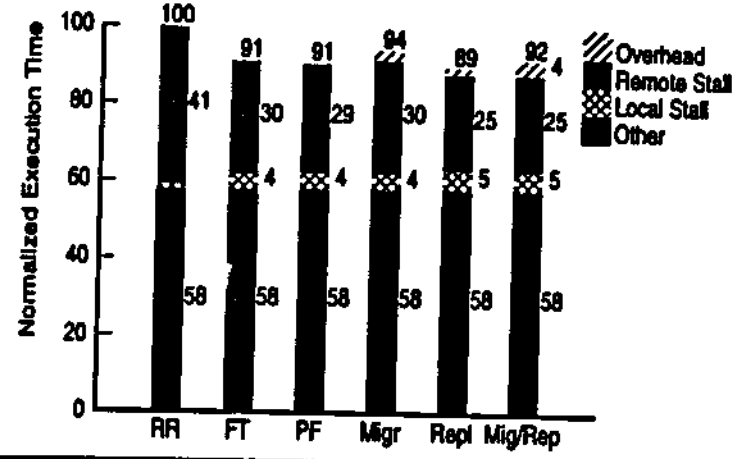


FIGURE 7: Execution time breakdown for the pmake workload. Only Kernel misses are considered for the different policies.

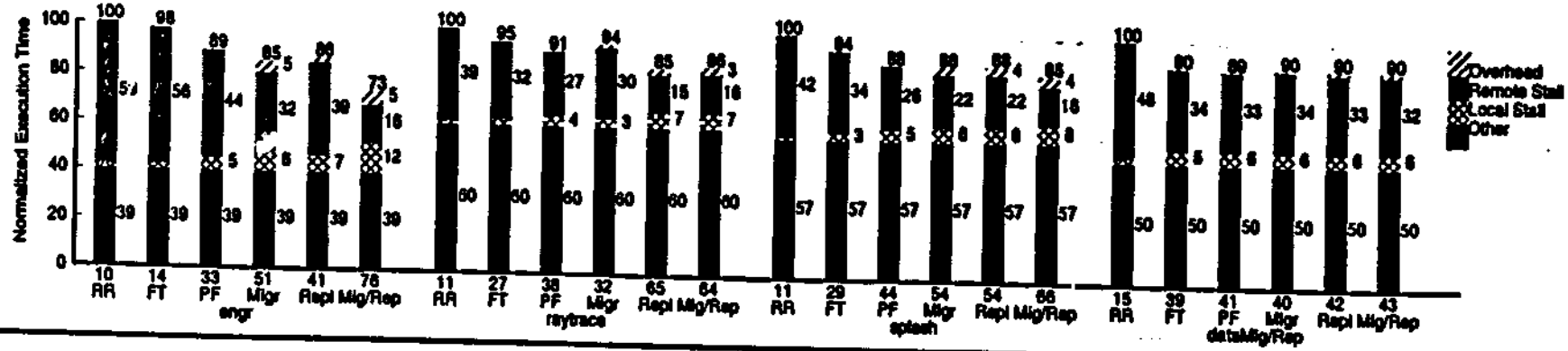


FIGURE 6: Breakdown of user execution time for various policies. There are six runs for each workload, Roundrobin (RR), First touch (FT), and Post-facto (PF), Migration-only (Migr), Replication-only (Repl), and the combined migration/replication policy (Mig/Rep). Each bar represents the execution time for a policy normalized w.r.t. the RR policy. Each bar shows cache-miss stall to local memory, cache-miss stall to remote memory, the overhead to migrate and replicate pages, and all other time. The percentage of misses to local memory is shown at the bottom of each bar.

Overheads

- Space taken by cntrs: 1 byte cntr, 8 proc, 4KB pages → 0.2%

  128 nodes ⇒ 3.1%

  Use sampling: half-sized cntrs: 1.6%

  Group processors (logically)

- Total latency of 1 operation: 500 μs
- Most ovhd from TLB flush → try to flush only those proc that have the mapping
- Lock overhead too; not pg copying itself
- Mem increase: 30%

- Performance better than static policies
- Not much impact for kernel data
- Sampled cache misses is OK metric
- TLB misses : too coarse a metric
- Varying parameters:
  - → Trigger threshold : best one depends on local/remote miss lat, OS overhead ...
  - → Sharing threshold: no impact ⇒ pages have clear behavior

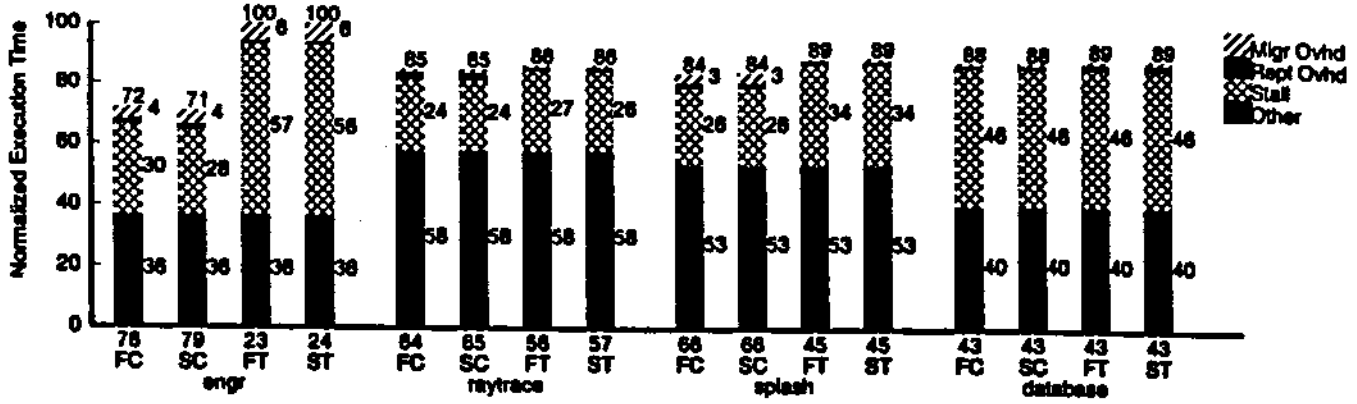


FIGURE 8: Performance impact of approximate information. There are 4 bars for each workload, Full cache (FC), Sampled cache (SC), Full TLB (FT), and Sampled TLB (ST). All sampled cases are sampled with ratio 1:10. Each bar shows the run time normalized to the run time for the Round robin case. The run time is broken down as User stall (local and remote), Overhead for replication and migration separately, and all other time. The percentage of misses made local is shown at the bottom of each bar.

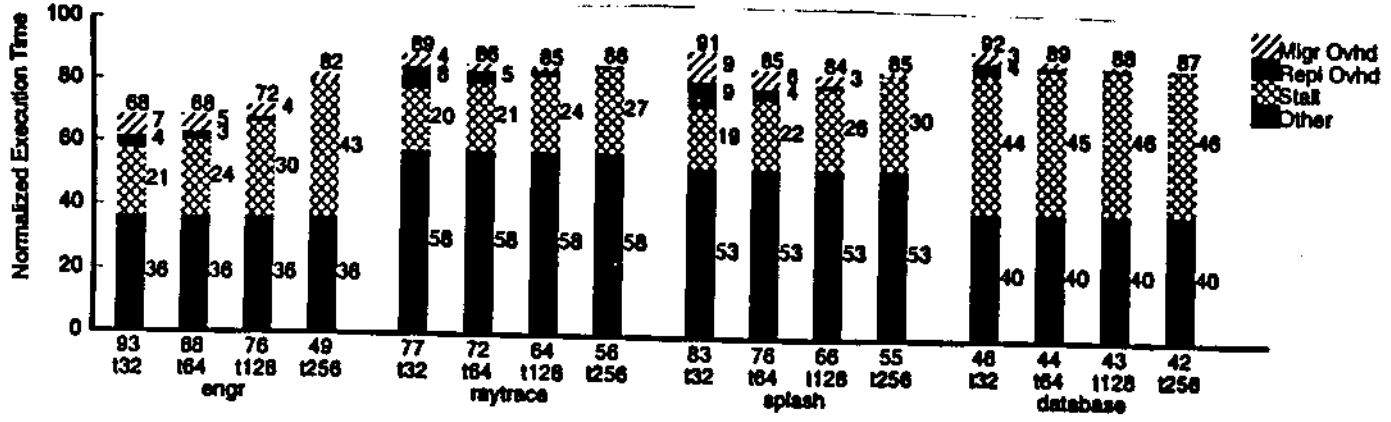


FIGURE 9: Variation in performance with the trigger threshold. Each workload is run with four trigger thresholds, 32, 64, 128, and 256. The sharing threshold is a quarter of the trigger threshold. Each bar shows the run time for a configuration normalized to the run time for the roundrobin case for that application. In each bar, we separately show User CPU, User stall, and overhead cost of replication and migration. The percentage of misses made local is shown at the bottom of each bar.