Analytical Cache Models with Applications to Cache Partitioning

G. Edward Suh, Srinivas Devadas, and Larry Rudolph

LCS, MIT

Motivation

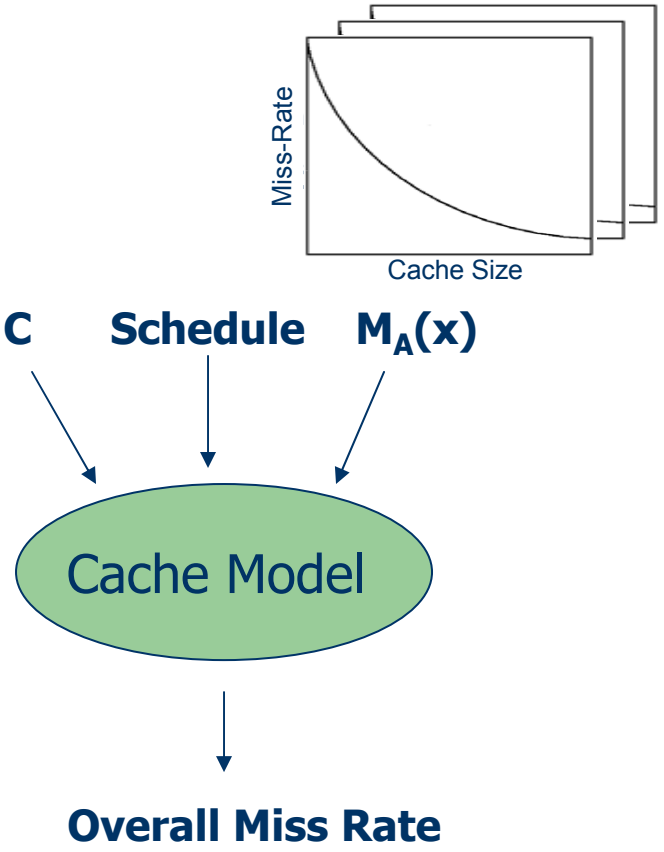
- Memory system performance is critical
- Everyone thinks about their own application
  - But modern computer systems execute multiple applications concurrently/simultaneously
  - Context switches cause cold misses
  - Simultaneous applications compete for cache space
- Caches should be managed more carefully, considering multiple processes
  - Explicit management of cache space => partitioning
  - Cache-aware job schedulers

Related Work

- Analytical Cache Models
  - Thiébaut and Stone (1987)
  - Agarwal, Horowitz and Hennessy (1989)
  - Both only focus on long time quanta
  - Inputs are hard to obtain on-line
- Cache Partitioning
  - Stone, Turek and Wolf (1992)
    - Optimal cache partitioning for very short time quanta
- Our Model & Partitioning
  - Work for any time quantum
  - Inputs are easier to obtain (possible to estimate on-line)

Our Multi-tasking Cache Model

- Input
  - C: Cache Size
  - Schedule: job sequences with time quantum (T_A)
  - $M_A(x)$: a miss rate as a function of cache size for Process A
- Output
  - Overall miss-rate (OMR) for multi-tasking



Assumptions

- The miss-rate of a process is a function of cache size alone, not time
- One MR(size) per application
  - Curve is averaged over application lifetime
  - In cases of high variance
    - Split the application into phases
    - One MR(size) per phase
  - Generated off-line (or on-line with HW support)
- No shared memory space among processes

Assumptions: Cont.

- Fully-associative caches
  - Extended to set-associative caches (memo 433)
  - The fully-associative model works for set-associative cache partitioning
- LRU replacement policy
- Time in terms of the number of memory references
  - The number of memory reference can be easily converted to real time in a steady-state

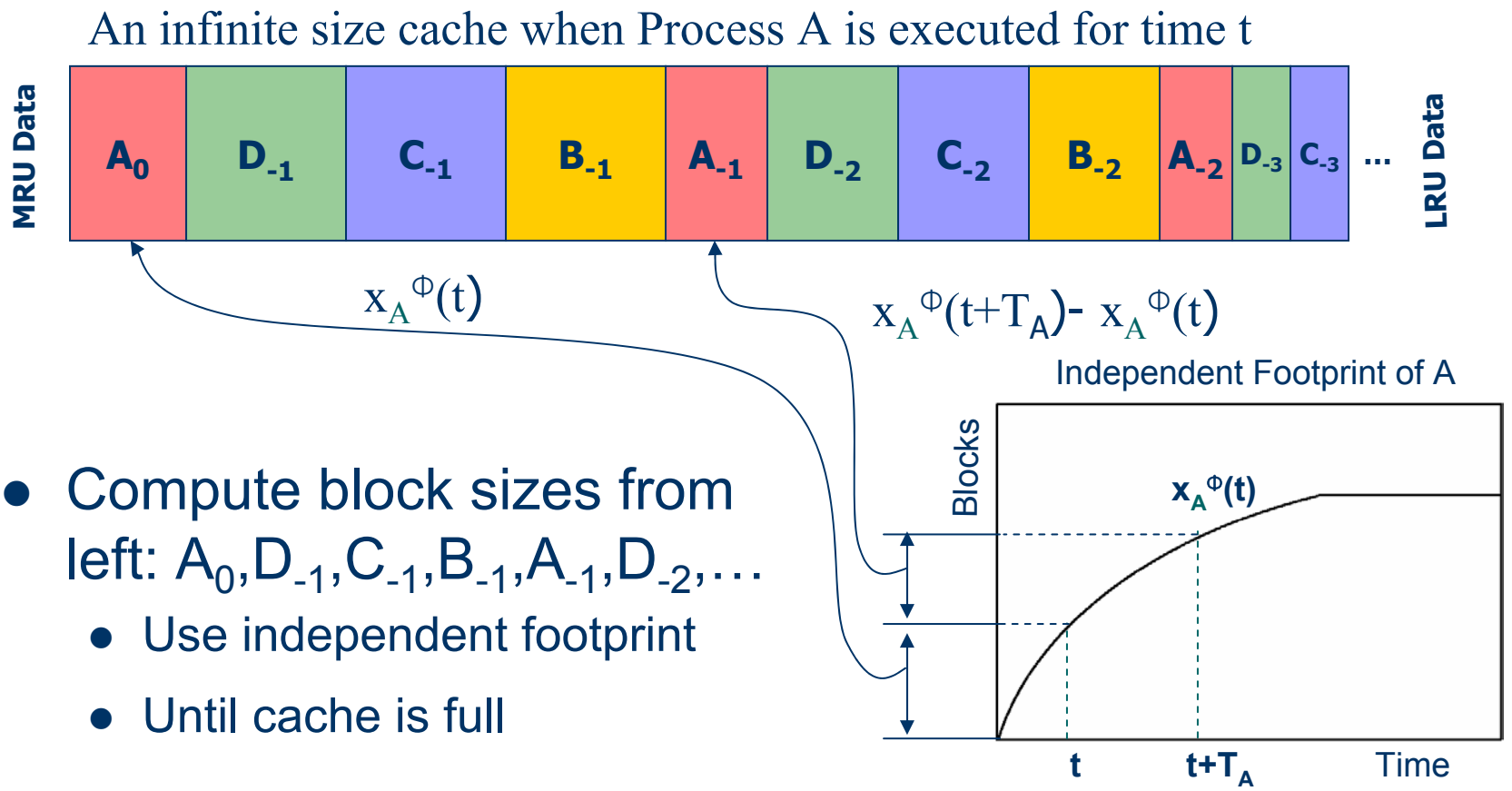
Independent Footprint $x_A^{\Phi}(t)$

- Independent footprint
  - The amount of data for Process A at time t starting from an empty cache, $x_A^{\Phi}(0) = 0$
  - Assume only one process executes
- Changes
  - If hit, $x_A^{\Phi}(t+1) = x_A^{\Phi}(t)$
  - If miss, $x_A^{\Phi}(t+1) = MIN[\ x_A^{\Phi}(t) + 1, C\ ]$
- If we approximate real value of $x_A^{\Phi}(t)$ with its expectation:
  - $E[x_A^{\Phi}(t+1)] = MIN[\ E[x_A^{\Phi}(t)] + P_A(t), C\ ]$
    $= MIN[\ E[x_A^{\Phi}(t)] + M_A(E[x_A^{\Phi}(t)]), C\ ]$

Dependent Footprint $x_A(t)$

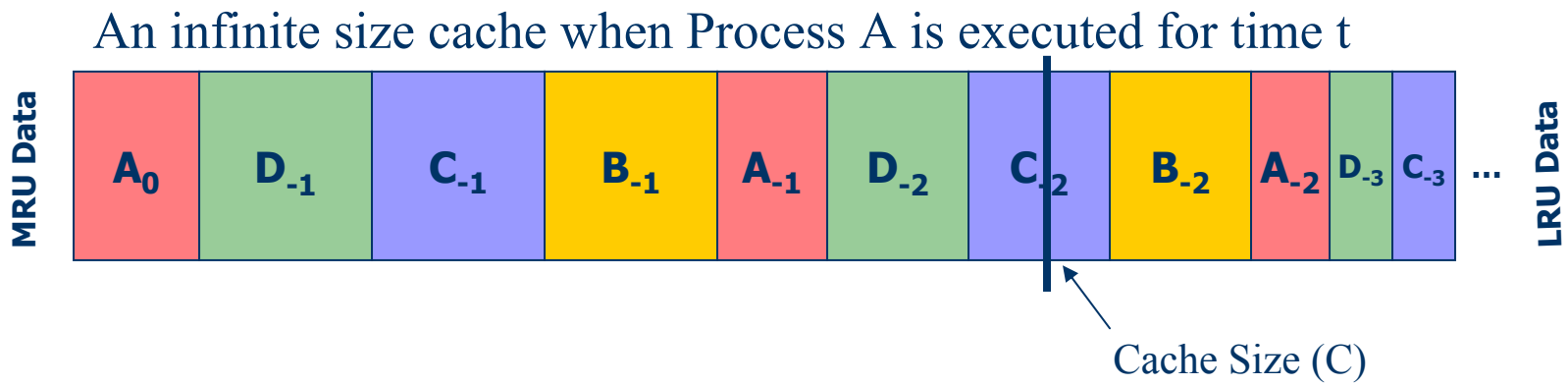
- Dependent footprint
  - The amount of data for Process A when multiple processes concurrently execute
  - Obtained from the given schedule and the independent footprint of all processes
- Example
  - Four processes: A, B, C, D
  - round-robin schedule: ABCDABCD…

Dependent Footprint $x_A(t)$: Cont.



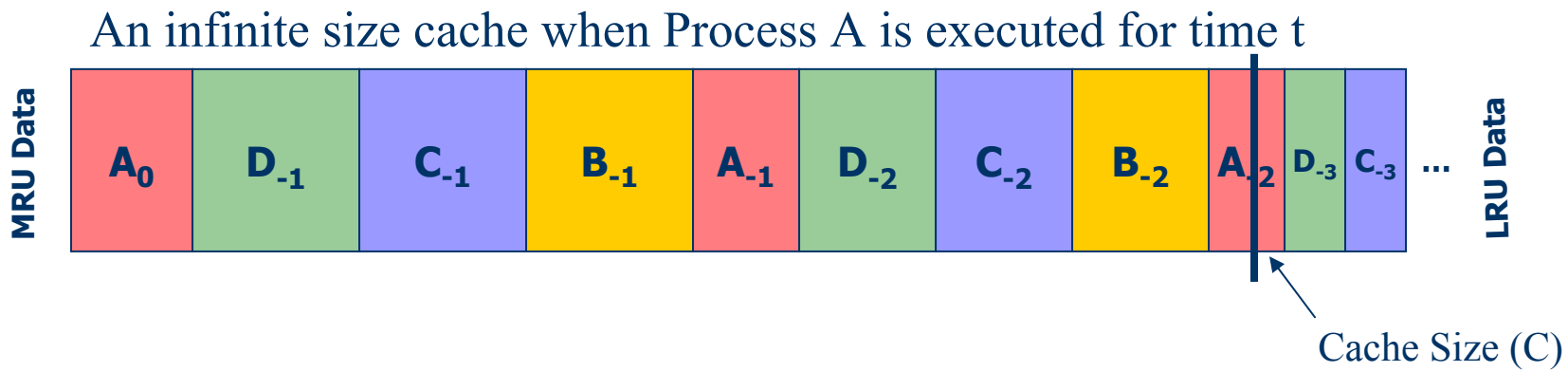
- Compute block sizes from left: $A_0, D_{-1}, C_{-1}, B_{-1}, A_{-1}, D_{-2}, \ldots$
  - Use independent footprint
  - Until cache is full

Dependent Footprint $x_A(t)$: Cont.



- **Case 1: dormant process' block is the LRU**
  - $x_A(t) = A_0 + A_{-1} = x_A^{\Phi}(t+T_A)$

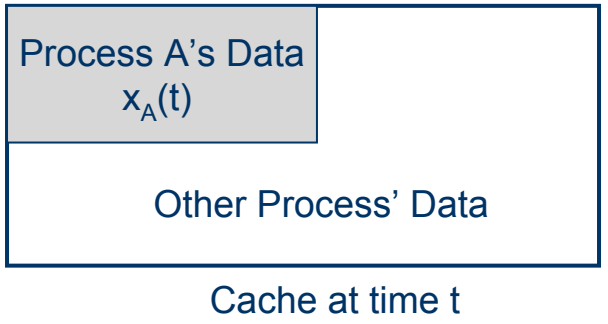
Dependent Footprint $x_A(t)$: Cont.



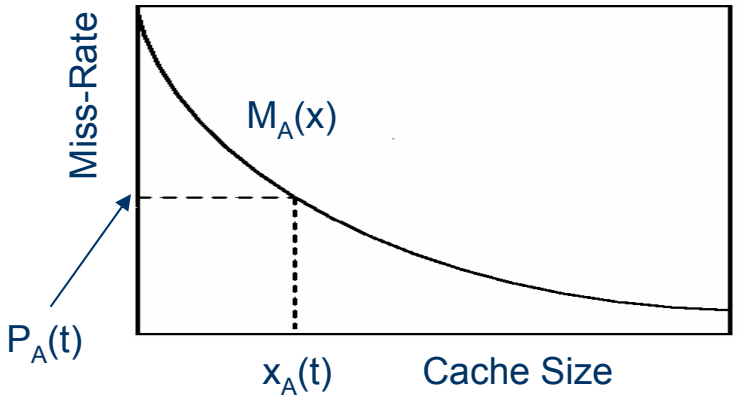
- Case 1: dormant process' block is the LRU
  - $x_A(t) = A_0 + A_{-1} = x_A^{\Phi}(t+T_A)$
- **Case 2: active process' block is the LRU**
  - **$x_A(t) = C-(D_0+C_0+B_0+D_{-1}+C_{-1}+B_{-1})$**
    $= C-x_D^{\Phi}(T_D)-x_C^{\Phi}(T_C)- x_B^{\Phi}(T_B)$

Computing the Miss Probability: $P_A(t)$

- Effective cache size
  - $x_A(t)$: The amount of data in a cache for process A at time t

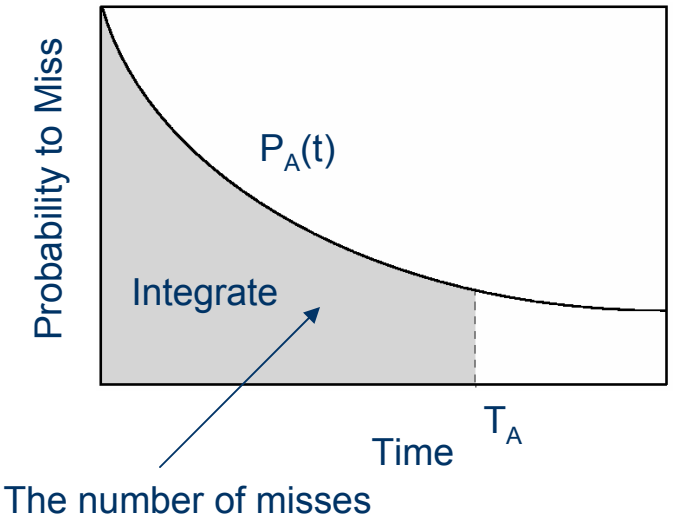


- The probability to miss at time t
  - $P_A(t) = M_A(x_A(t))$



Estimating Miss-Rate

- Miss-rate of Process A
  - In a steady-state, all time quanta of Process A are identical
  - Time starts (t=0) at the beginning of a time quantum
  - $$mr_A = \frac{1}{T_A}\int_0^{T_A} P_A(t)dt$$



- Overall miss-rate (OMR)
  - Weighted sum of each process' miss-rate

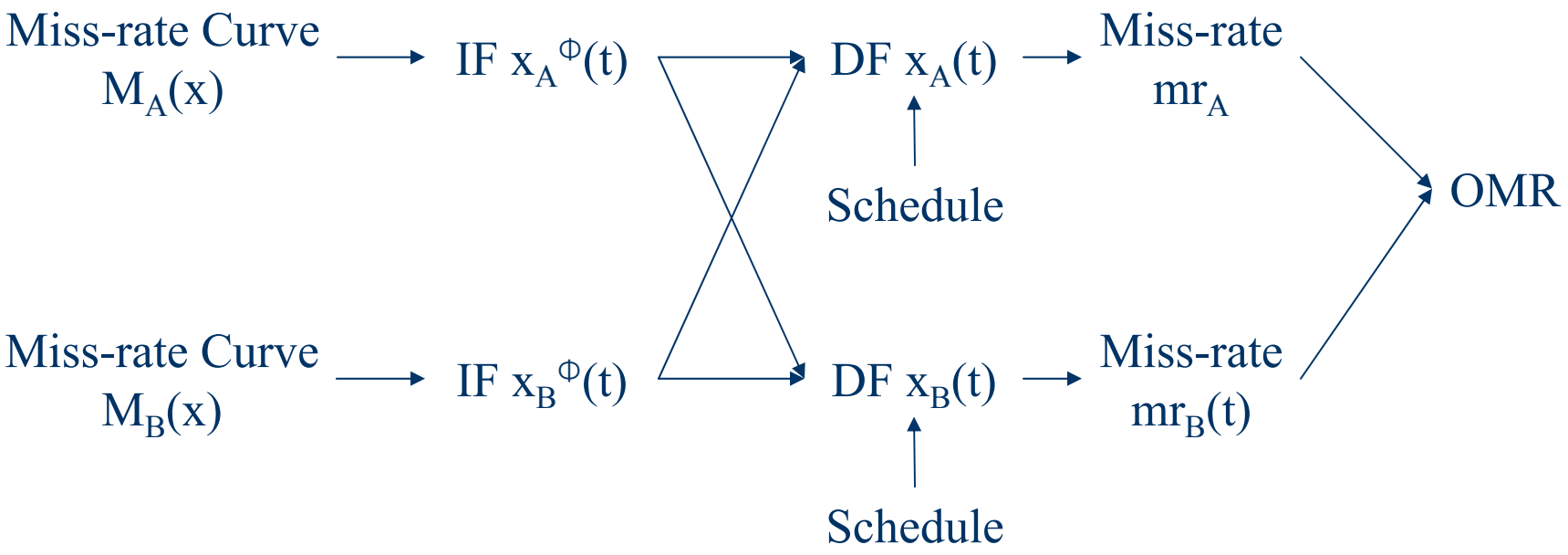
Model Summary

$E[x_A^{\Phi}(t+1)] = E[x_A^{\Phi}(t)] + M_A(x_A^{\Phi}(t))$

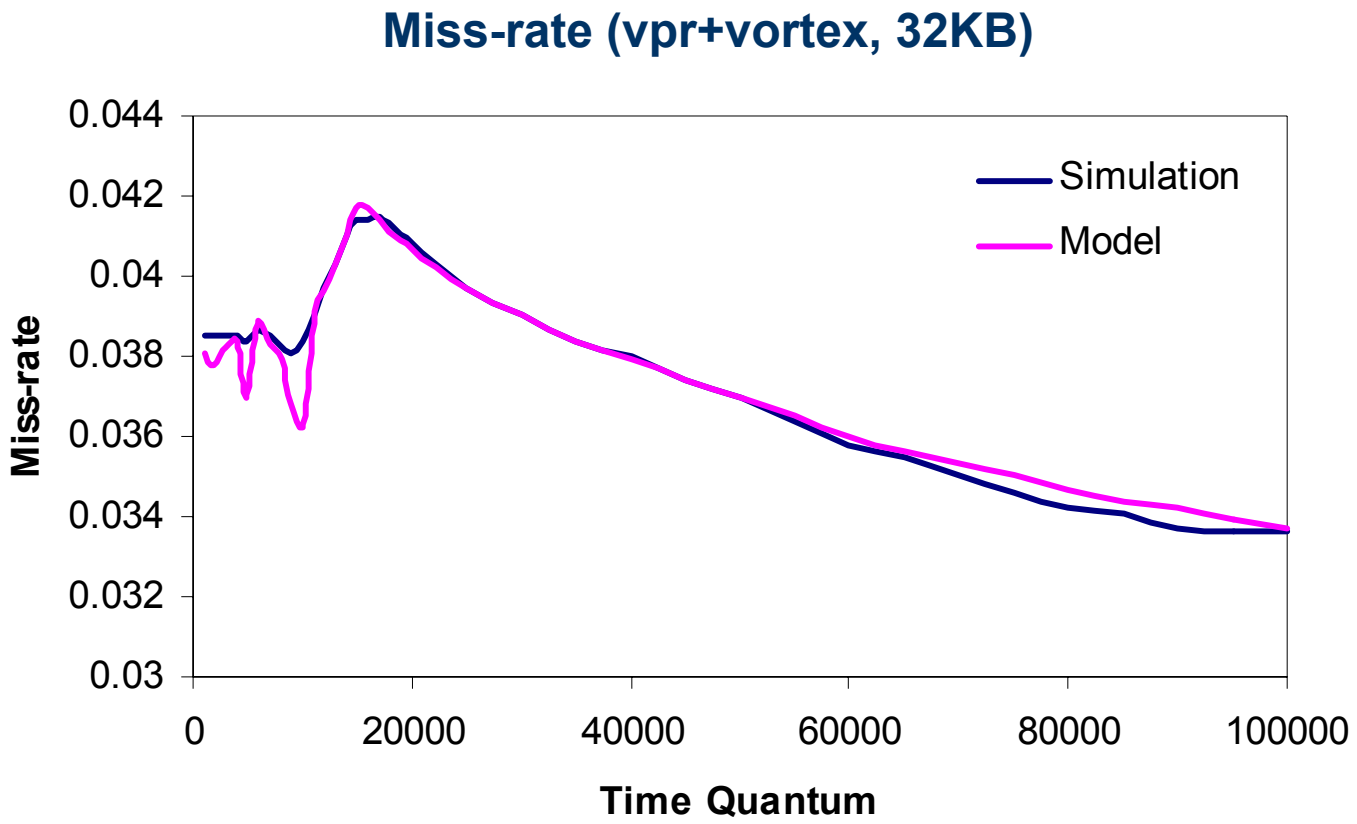
Cache snapshot

$\frac{1}{T_A}\int_0^{T_A} M_A(x(t))dt$

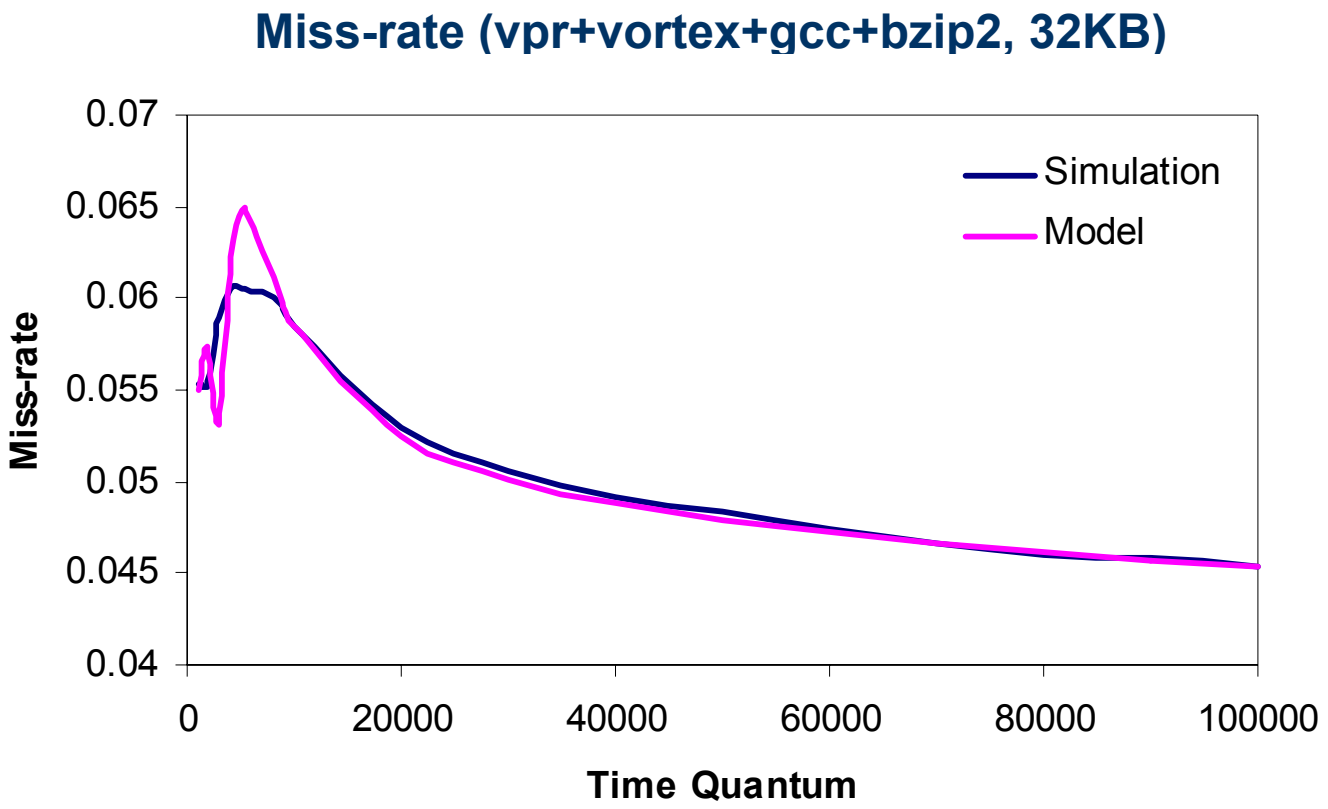
$\frac{1}{T_{sum}}\sum_{i=1}^{N} mr_i \cdot T_i$



Model vs. Simulation: 2 Processes



Model vs. Simulation: 4 Processes



Cache Partitioning

- Time-sharing degrades the cache performance significantly for some time quanta
  - Due to dumb allocation by LRU policy
  - Could be improved by explicit cache partitioning
- Specifying a partition
  - Dedicated Area (D_A)
    - Cache blocks that only Process A can use
  - Shared Area (S)
    - Cache blocks that any process can use while it is active

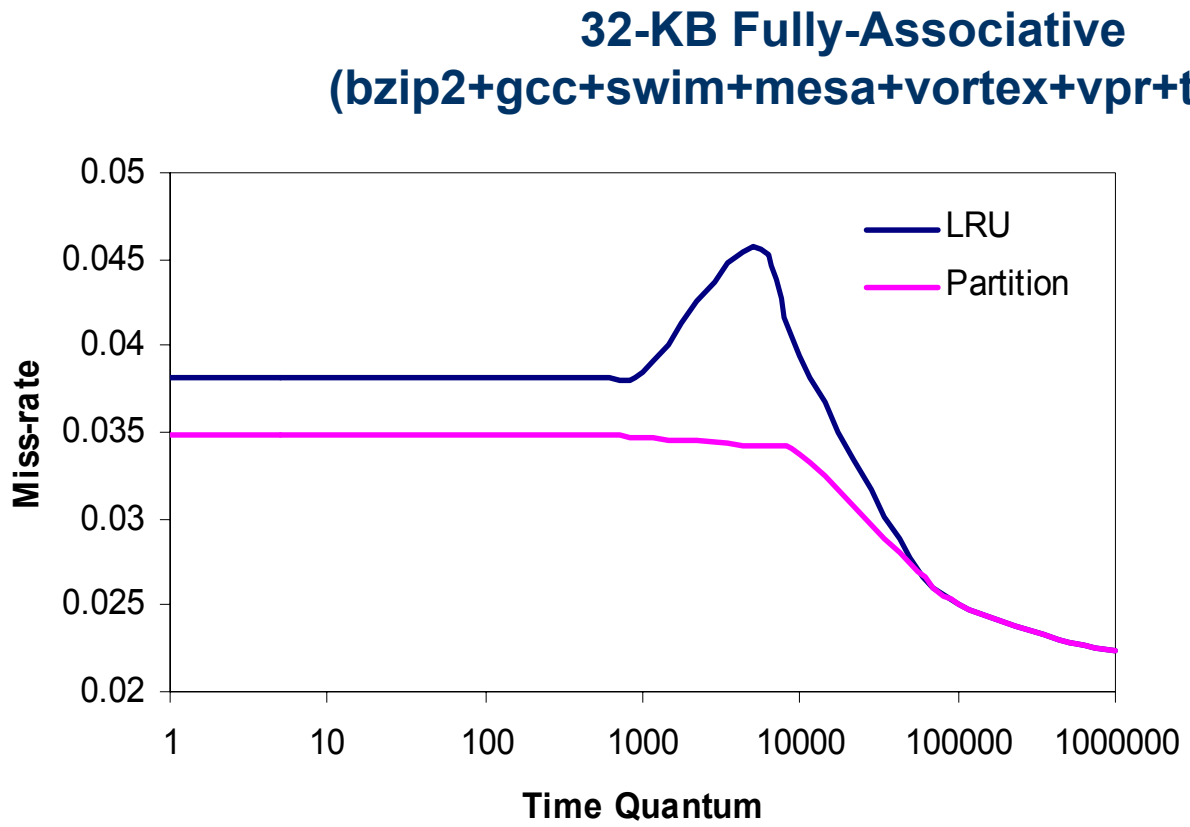
Strategy

- Off-line profiling of MR(size) curves
  - One for each phase
  - Independent of other processes
  - Can also be obtained on-line with HW support
- On-line partitioning
  - Partitioning decision based on the model
  - Modify the LRU policy to partition the cache

Optimal Cache Partition

- Dedicated areas (D_A) specify the initial amount of data for each process
  - $x_A(0) = D_A$
- Shared (S) and dedicated (D_A) areas specify the maximum cache space for each process
  - $C_A = D_A + S$
- The model can estimate the miss-rate for a given partition
- Use a gradient based search algorithm

Simulation Results: Fully-Associative Caches

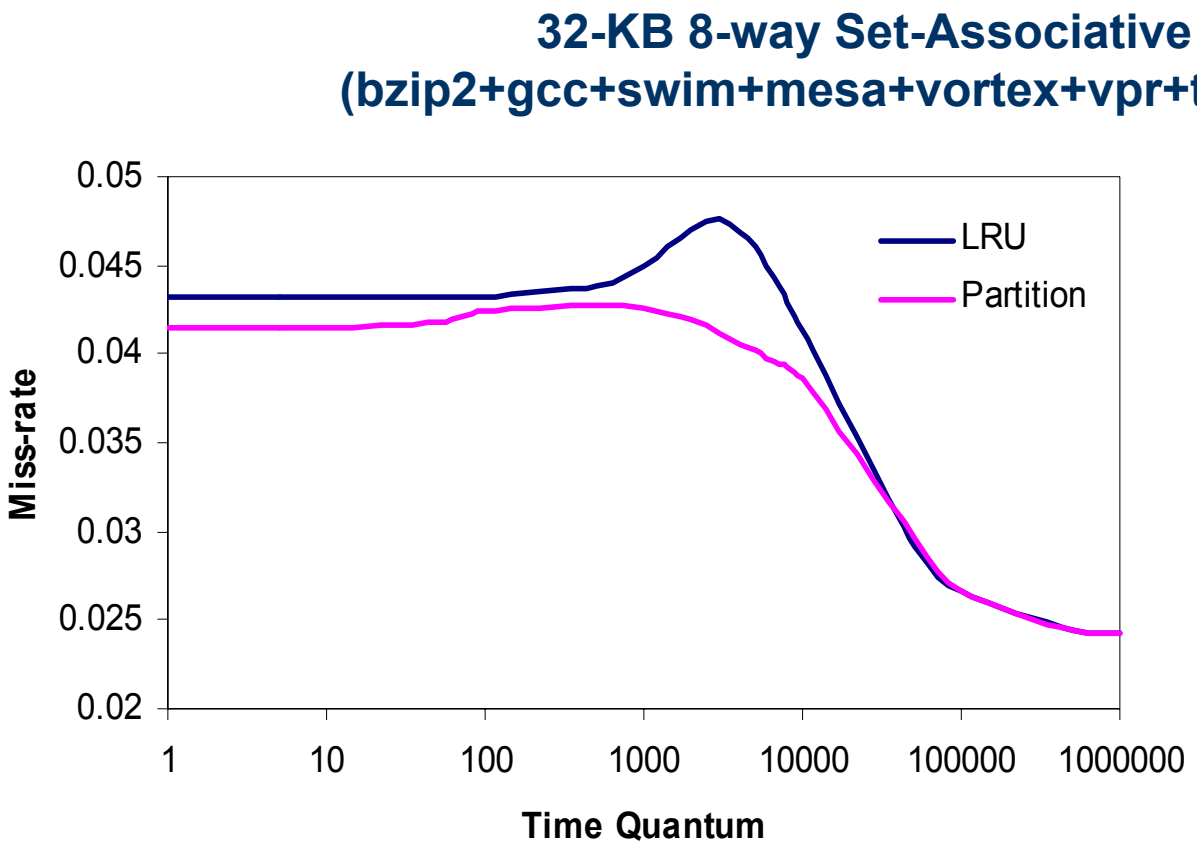


- 25% miss-rate improvement in the best case
- 7% improvement for short time quanta

From Full to Partial Associative

- Use the fully-associative model and curves to determine D_A, S
- Modify the LRU replacement policy to partition
  - Count the number of cache blocks for each process (X_A)
  - Try to match X_A to the allocated cache space
  - Replacement (Process A active)
    - Replace Process A's LRU block if $X_A \geq D_A + S$
    - Replace Process B's LRU block if $X_B \geq D_B$
    - Replace the standard LRU block if there is no over-allocated process
- Add a small victim cache (16 entries)

Simulation Results: Set-Associative Caches



- 15% miss-rate improvement in the best case
- 4% improvement for short time quanta

Summary

- Analytical cache model
  - Very accurate, yet tractable
  - Works for any cache size and time quanta
  - Applicable to set-associative cache partitioning
- Applications
  - Dynamic cache partitioning with on-line/off-line approximations of miss-rate curves
  - Various scheduling problems