

Energy- Optimal Buffer Cache Replacement

Jianhui Yue Yifeng Zhu Zhao Cai

*Department of Electrical and Computer Engineering, University of Maine
Orono, USA*

jyue, zhu, zcai@eece.maine.edu

I. INTRODUCTION

With speed gap between the processor and disk is increasingly wide, much memory are used as the buffer cache to reduce the gap. Scuh large capacity of memory can consume much energy than processor and even reaches 40% total computer machine's power consumption. In order to reduce the memory power consumption, the mulitple power states are introduced in the morden memory chip. In the lower power state, some parts of memory comptonets are powered off without losssing data and power saving is achieved. Howerer, power state transition from lower to higher invloves both performance and energy penalty. In order to reduce the negative effects of power state transition, the dynamical power down policy is widely adopted at existing research works. In the dynamic power down policy, the memory chip is transitioned to low power state when it is free for a threshold time, which can be predefined one or varied in response to workload dynamics. In this report, we focus on the buffer cache memory consumption especailly for the I/O server. In the I/O server, the data requests come from clients through network and response data are transferred from server's buffer cache to the network adaptor through DMA. In the case of data miss in the buffer cache, data is retrieved from the disk to buffer cache through DMA. In the I/O server, the DMA operations are heavily used. The morden DMA controller can overlap multiple concurrent DMA operations involved with one memory chip. In the overlapping mode, the concurrent DMA operations can enjoy the free ride and achieve the energy saving. The maximum of overlapping is limited the ratio of memory chip bandwidth to bus bandwidth. For the convenience of discussion, each potential DMA overlapping opration over a memory chip is abstracted as a slot. Each slot has its corresponding latest DMA operation end time which governs the corresponding memory chip power state transition.

II. ASSUMPTIONS

Assumptions are listed as follows.

- α : predefined power down threshold
- θ_m : memory access time for one hit request
- θ_{dm} : disk and memory access time for one missed request
- No power state transition panely

- No resource confliction (no queue)

III. NOTATIONS

Notations are listed as follows.

- a_i : The i th request.
- C_k : The cache content for the k th memory chip is set of block numbers resided in the k th memory chip. If request $a_i \in C_k$, this request hit the k th memory chip. Otherwise, no memory access occurs at this chip or replacement occurs at this chip.
- CT_k : The slots' completion time for the k th memroy chip. The modern DMA controller can support overlapping mulitple DMA operations and each DMA operation in the specified memory chip is abstracted as slot. For example, the DMA controller has capabilitiy to overlap four DMA operations to one memory chip and this memory chip has four slots. Each slot's completion time is recorded at CT_k .
- $CT_k.S_{free}$: The slot index for the free one.
- $CT_k.S_{busy}$: The slot index for the lastest busy one.
- r : The memory chip index for the replacement.
- h : The memory chip index for the access hit.
- $f_k(C_k, CT_k, r, i)$: The power up duration for k th memory chip when the i th request arrives .
- $F(i)$: The summation of all chips' active time.
- C : The contents of all memory chips $\bigcup C_k$.

IV. OPTIMAL ALGORITHM

$$f_k(C_k, CT_k, r, i+1) = \begin{cases} f_k(C_k, CT_k, r, i) & CT_k[S_{busy}] > \alpha \wedge r! = k \wedge a_{i+1} \notin C_k \wedge r \neq k \quad (1) \\ f_k(C_k, CT_k, r, i) + 1 & CT_k[S_{busy}] \leq \alpha \wedge a_{i+1} \notin C_k \wedge r \neq k \quad (2) \\ f_k(C_k, CT_k!, r, i) + 1 & a_{i+1} \in C_k \wedge CT_k![CT_k!.S_{free}] = (i+1) + \theta_m \\ & \wedge CT_k!.S_{busy} = CT_k!.S_{free} \\ & \wedge CT_k!.S_{free} = \min(CT_k!) \wedge CT_k = CT_k! \wedge r \neq k \quad (3) \\ f_k(C_k!, CT_k!, r, i) + 1 & a_{i+1} \in C_k \in repl(C_k!, a_{i+1}) \\ & \wedge CT_k![CT_k!.S_{free}] = (i+1) + \theta_{dm} \wedge CT_k!.S_{busy} = CT_k!.S_{free} \\ & \wedge CT_k!.S_{free} = \min(CT_k!) \wedge CT_k = CT_k! \wedge r = k \quad (4) \end{cases}$$

$$F(i+1) = \begin{cases} \sum_{k=1, k \neq h}^n f_k(C_k, CT_k, 0, i) + \min_{C_{h'}} f_h(C_h, CT_{h'}, 0, i) & a_{i+1} \in C \wedge a_{i+1} \in C_h \quad (5) \\ \min_{C_k!, C_k!, r!, r! \neq 0} (\sum_{k=1}^n f_k(C_k!, CT_k!, r!, i)) & a_i \notin C \quad (6) \end{cases}$$

- Equation 1 : It is in the power down state and the chip's active duration is not incremented.
- Equation 2 : It is in the idle state and chip's active duration is incremented.
- Equation 3 : This chip contains request a_{i+1} data, the chip's slot information is updated and active duration is incremented.
- Equation 4 : Replacement occurs at this chip for the request a_{i+1} , the chip's slot information is updated and active duration is incremented.
- Equation 5 : Request a_{i+1} data hit buffer cache.
- Equation 6 : Request a_{i+1} data misses buffer cache and replacement occurs at memory chip $r!$. The $r!$ with the value of 0 means no replacement happens in any memory chip. The energy-optimal buffer cache replacement algorithm try its best to find the replacement with the minimal summation of all chips' active duration.

V. APPROXIMATION ALGORITHM : LOOKING FORWARD NEXT ONE DETERMINISTIC MISS REQUEST

A. Choose Victim Chip

Fig. 1. Requests distribution over memory chip j

The request a_i is the miss one which can be determined from current cache content and request sequence $\{a_k\}$. In the approximation replacement algorithm, we only consider the missed request since hit requests will not change the buffer cache content and three locations were determined by this approximation replacement algorithm. For the missed request a_i , we have following notations.

- $P_j(a_i)$: The chip j completion time for the request a_i which is the closest to request a_i and arrives immediately before request a_i . The completion time does take the overlapping into account. Such request is referred as a_i 's leader over chip j.
- $N_j(a_i)$: The arrival time of the request to the chip j which is the closest to request a_i and arrives immediately after request a_i . Such request is referred as a_i 's follower over chip j.
- $T(a_i)$: The time when the request a_i arrives.
- $E(t)$: The energy consumption of chip being active for t time unit considering the power down.

If we choose victim from chip j, its energy penalty is $E(T(a_i) - P_j(a_i)) + E(N_j(a_i) - T(a_i)) - E(N_j(a_i) - P_j(a_i))$ when request a_i miss cache. Choosing victim from this chip, its idle time duration $[P_j(a_i), N_j(a_i)]$ is broken into two smaller durations $[P_j(a_i), T(a_i)]$ and $[T(a_i), N_j(a_i)]$. Victim is chosen from chip whose energy penalty is the minimal. This requires looking forward a certain number of requests covering $\max N_j(a_i)$.

B. Choose Victim Block from Victim Chip

Fig. 2. Requests distribution over memory chip j

After determining the victim chip, we begin to choose victim block from this chip.

- $Q_j(k)$: The chip j completion time for the request to access each block in the memory chip j ordered by the time.

- $\Delta_j(k) : Q_j(k) - Q_j(k - 1)$

- $\Delta_{j1}(k) : Q_j(k) + T_{disk} - Q_j(k - 1)$

Two Methods:

- To replace the data block corresponding to k which has minimal of $\Delta_j(k)$. Use Belady to break tie.
- To replace the data block corresponding to k which has minimal of $\Delta_{j1}(k)$. Use Belady to break tie