

(Yet another) proof of optimality for MIN replacement

Pierre Michaud

October 30, 2007

The MIN cache (or page) replacement policy proposed by Belady [1] is an optimal off-line replacement policy. Intuition tells that MIN must be optimal. The first published proof of optimality was very detailed [4]. Some other proofs have been proposed since [2, 5, 3, 6]. The proof sketched in [5] is an extremely concise version of Mattson et al.'s proof but is difficult to understand. The proof we propose hereafter is basically the same as [4] and [5], but its level of detail is intermediate.

Problem statement

Let us consider a set of M blocks numbered from 1 to M , and a finite block sequence $(b(t))$, with $t = 1, 2, 3, \dots$ and $b(t) \in [1, M]$. A time t , the *cache* is a set $S(t) \subset [1, M]$. The cache size $|S(t)| = N < M$ is fixed and constant. Without loss of generality, we assume $S(1) = [1, N]$.

If $b(t) \in S(t)$, this is a *hit*. Otherwise, if $b(t) \notin S(t)$, this is a *miss*, and a *victim* block $v(t)$ is evicted from the cache to make room for block $b(t)$:

$$S(t+1) = (S(t) - \{v(t)\}) \cup \{b(t)\}$$

By convention, we define $\{v(t)\} = \emptyset$ when $b(t) \in S(t)$.

We define $T_j(t)$ as the earliest $\tau > t$ such that $b(\tau) = j$, i.e., it is the next reference time for block j after time t . By convention, $T_j(t) = \infty$ if there is no reference to block j after time t . The MIN replacement policy evicts the block whose next reference is furthest in the future :

$$v(t) = \operatorname{argmax}_{j \in S(t)} \{T_j(t)\}$$

In order to prove that MIN is an optimal replacement policy (i.e., it minimizes the number of misses), we are going to show that, for any replacement policy R whose first departure from MIN replacement is at time τ , it is possible to derive a policy R' that does not generate more misses than R and whose first departure from MIN replacement is at a time strictly greater than τ . By applying this procedure repeatedly, we can transform any policy R into a MIN policy without increasing the number of misses, which will prove the optimality of MIN.

Proof

Let us consider a replacement policy R , different from MIN, that victimizes blocks $\{v(t)\}$ and generates cache states $S(t)$. From R , we are going to derive a policy $R' \neq R$ victimizing blocks $\{v'(t)\}$ and generating cache states $S'(t)$. Let τ_1 be the earliest miss time such that $v(\tau_1) \neq v_m$ where

$$v_m = \operatorname{argmax}_{j \in S(\tau_1)} \{T_j(\tau_1)\}$$

That is, τ_1 is the earliest departure of policy R from MIN. For $t < \tau_1$, policy R' imitates R , i.e., $\{v'(t)\} = \{v(t)\}$. Consequently, $S'(\tau_1) = S(\tau_1)$. At time τ_1 , instead of victimizing block $v(\tau_1)$, policy R' victimizes block $v'(\tau_1) = v_m$. Hence the first departure of policy R' from MIN happens at a time strictly greater than τ_1 . We define sets $D(t)$ and $D'(t)$ as follows :

$$\begin{aligned} D(t) &= \{i \in S(t) / i \notin S'(t)\} \\ D'(t) &= \{i \in S'(t) / i \notin S(t)\} \end{aligned}$$

We have

$$\begin{aligned} D(\tau_1 + 1) &= \{v_m\} \\ D'(\tau_1 + 1) &= \{v(\tau_1)\} \end{aligned}$$

We define sequence (τ_k) as follows. For $k \geq 1$,

$$\tau_{k+1} = T_{v(\tau_k)}(\tau_k)$$

For $t > \tau_1$ and **as long as** $b(t) \neq v_m$ **and** $v(t) \neq v_m$, policy R' behaves as follows. Initially, $k = 1$. At time $t = \tau_k + 1$, we have

$$\begin{aligned} D(t) &= \{v_m\} \\ D'(t) &= \{v(\tau_k)\} \end{aligned} \quad (1)$$

We iterate the following steps.

Step k : for $\tau_k < t < \tau_{k+1}$

As $b(t) \notin D(t)$, a hit by R implies a hit by R' . Equations (1) remain true for $t + 1$. As $b(t) \neq b(\tau_{k+1}) = v(\tau_k)$, i.e., $b(t) \notin D'(t)$, a miss by R implies a miss by R' . In this case, as $v(t) \notin D(t)$ (from the assumption $v(t) \neq v_m$), $v(t) \in S'(t)$ and we can choose $v'(t) = v(t)$ (both policies evict the same block) and equations (1) remain true for $t + 1$.

End of Step k : $t = \tau_{k+1}$

We have $b(t) = v(\tau_k)$. Policy R' hits while policy R misses. Block $v(\tau_k)$ replaces block $v(\tau_{k+1})$ in $S(t)$ and block $v(\tau_{k+1})$ replaces block $v(\tau_k)$ in $D'(t)$: we have $D'(t+1) = \{v(\tau_{k+1})\}$. We go to the next step, $k \leftarrow k + 1$, and equations (1) remain true for $t + 1$. At the end of step k , policy R has generated exactly k extra misses compared to R' .

R and R' rejoin

The procedure described above ends with sequence $(b(t))$ or when the assumption $b(t) \neq v_m$ and $v(t) \neq v_m$ becomes false. If the case $v(t) = v_m$ occurs first, then $S'(t + 1) = S(t + 1)$, and from now on, R' follows R exactly. Otherwise, if the case $b(t) = v_m$

occurs first (at time $T_{v_m}(\tau_1)$), this is a hit for R and a miss for R' (equations (1)). Policy R' evicts block $v(\tau_k)$, so that $S'(t + 1) = S(t + 1)$. Then R' is identical to R . It should be noted that the definition of v_m implies $\tau_2 < T_{v_m}(\tau_1)$. So when $b(t) = v_m$, we must have $k \geq 2$. Before R' misses on v_m , R has generated at least one extra miss compared to R' . Hence R' does not generate more misses than R .

In any case, policy R' does not generate more misses than R . By repeating this procedure, we derive a policy R'' from R' , and so on, until obtaining the MIN policy for the whole sequence. As we made no assumption on policy R , this proves the optimality of MIN.

References

- [1] L.A. Belady. A study of replacement algorithms for a virtual-storage computer. *IBM Systems Journal*, 5(5):78–101, 1966.
- [2] E.G. Coffman and P.J. Denning. *Operating Systems Theory*. Prentice-Hall, 1973.
- [3] A. Cohen and W.A. Burkhard. A proof of the optimality of the MIN paging algorithm using linear programming duality. *Operations Research Letters*, 18(1):7–13, August 1995.
- [4] R.L. Mattson, J. Gecsei, D.R. Slutz, and I.L. Traiger. Evaluation techniques for storage hierarchies. *IBM Systems Journal*, 9(2):78–117, 1970.
- [5] L.A. McGeoch and D.D. Sleator. A strongly competitive randomized paging algorithm. *Algorithmica*, 6:816–825, 1991.
- [6] B. Van Roy. A short proof of optimality for the MIN cache replacement algorithm. *Information Processing Letters*, 102(2):72–73, April 2007.