Trimodal analysis

Brandon Chen

November 29, 2016

1 Trimodal

Trimodal: application scans three arrays simultaneously.

- x_1: size of array 1
- x_2: size of array 2
- x_3: size of array 3
- p_1: probability of access array 1
- p_2: probability of access array 2
- p_3: probability of access array 3
- $d_1 = \frac{x_1}{p_1}$: reuse distance for data in array 1
- $d_2 = \frac{x_2}{p_2}$: reuse distance for data in array 2
- $d_3 = \frac{x_3}{p_3}$: reuse distance for data in array 3
- $ed = pd_1 + p_2d_2 + p_3d_3$: expected reuse distance, equals the size of working set
- S: cache size
- m: miss rate

The reuse distance distribution would look like:

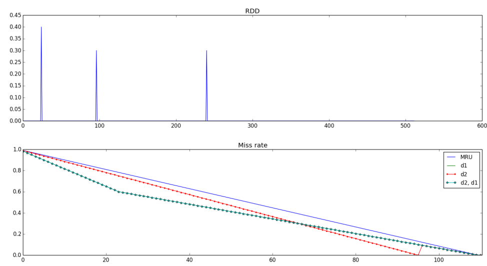

Figure 1: reuse distance distribution

Nathan proposed an awesome cache modeling paper [1], we derive the relationship between cache size and miss rate based on the paper's idea:

$$S = \sum a[H(a) + E(a)] \tag{1}$$

This means cache size equals the average lifetime. This may not seem obvious. Consider every access starts a new lifetime. So the average life time equals to the expected interval between two consecutive access on the same cache line, which is S.

1.1 evict at 0 (MRU)

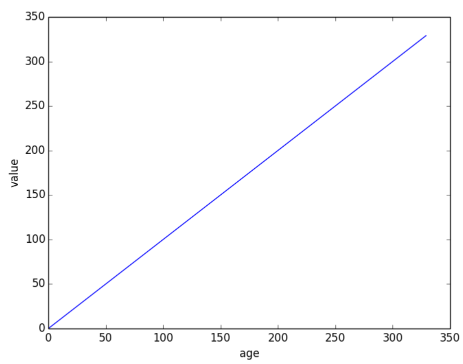

Figure 2: policy: mru

	0	d1	d2	d3
hit		$p_1(1-m)$	$p_2(1-m)$	$p_3(1-m)$
evict	m			

Table 1: events distribution when $0 < S < d1$

$$m = 1 - \frac{S}{ed} \tag{2}$$

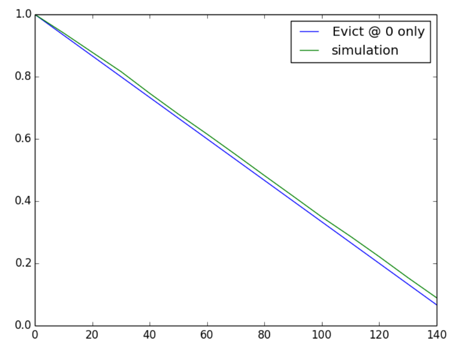

Figure 3: simulation result

1.2 evict at d1

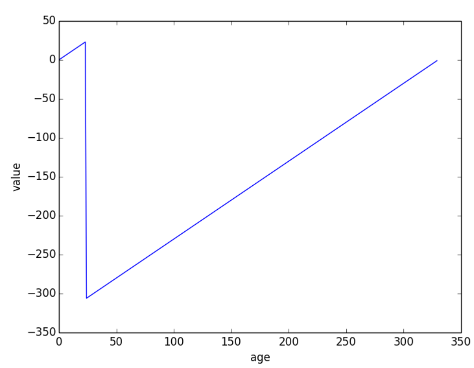

Figure 4: policy: evict candidate at age d1

	0	d1	d2	d3
hit		$p_1 x$		
evict	1-x	$(1-p_1)x$		

Table 2: events distribution when $0 < S < d1$

events	1	d1	d2	d3
hit		p1	$\frac{p_2}{1-p_1}(1-m)$	$\frac{p_3}{1-p_1}(1-m)$
evict		m		

Table 3: events distribution when $S > d_1$

$$m = \begin{cases} 1 - p_1 \frac{S}{d_1}, & \text{for } 0 \leq S \leq d_1 \\ (1-p_1)\frac{ed-S}{ed-d_1}, & \text{for } d_1 \leq S \leq ed \end{cases}$$

1.3 evict at d2, then 0

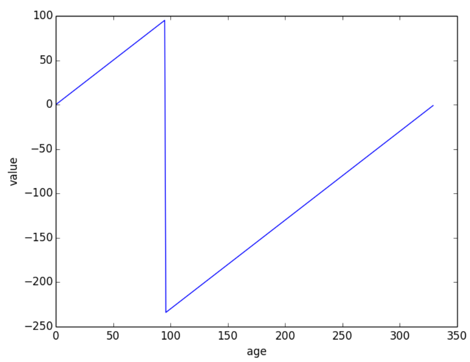

Figure 5: replacement policy in value function

Let x denote fraction of candidates who make it to the age of d1.

events	1	d1	d2	d3
hit		p_1x	p_2x	
evict	1-x		p_3x	

Table 4: events distribution when $S < d_2$

As cache size grow to some critical point, eviction distribution will change to the following.

events	0	d1	d2	d3
hit		p_1	p_2	$p_3 - m$
evict			m	

Table 5: events distribution when $S > d_2$

$$m = \begin{cases} 1-(1-p_3)\frac{S}{p_1d_1+(1-p_1)d_2}, & \text{for } 0 \leq S \leq p_1 * d_1 + (1-p_1) * d_1 \\ \frac{ed-S}{d_3-d_2}, & \text{for } d_2 < S < ed \end{cases}$$

1.4 Evict at d2, d1, then 0

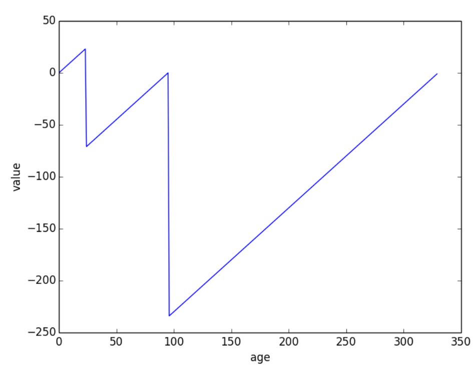

Figure 6: replacement policy graph

Let x denote fraction of candidates who make it to the age of d1.

	0	d1	d2	d3
hit		$p_1 x$		
evict	1-x	$(1-p_1)x$		

Table 6: events distribution when $0 < S < d1$

	0	d1	d2	d3
hit		p_1	$\frac{p_2}{1-p_1}x$	
evict		$1-x-p_1$	$\frac{p_3}{1-p_1}x$	

Table 7: events distribution when $d_1 < S < d_2$

	0	d1	d2	d3
hit		p_1	p_2	$p_3 - m$
evict			m	

Table 8: events distribution when $S > d_2$

$$m = \begin{cases} 1 - p_1 \frac{S}{d_1}, & \text{for } 0 < S < d_1 \\ 1 - p_1 - p_2 \frac{S-d_1}{(1-p_1)(d_2-d_1)}, & \text{for } d_1 \leq S \leq d_2 \\ \frac{ed-S}{d_3-d_2}, & \text{for } d_2 < S < ed \end{cases}$$

2 Solving MDP with analytical approach

First we want to have incremental relationship in Δ space: $\Delta V_a = V_a - V_0$.

According to the value iteration update algorithm:

$$\begin{aligned} V_a' &= h_a(1+V_0) + e_a V_0 + l_a V_{a+1} \\ V_0' &= h_0(1+V_0) + e_0 V_0 + l_0 V_1 \end{aligned} \tag{3}$$

When converge, we have $V_a' - V_0' = V_a - V_0$, that is:

$$\begin{aligned} \Delta V_a &= V_a' - V_0' \\ &= (1-h_a-e_a)\Delta V_{a+1} - (1-h_0-e_0)\Delta V_1 + h_a - h_0 \\ &= l_a \Delta V_{a+1} - l_0 \Delta V_1 + h_a - h_0 \end{aligned} \tag{4}$$

Here l_a denotes the leftover probability at age a, that is, $l_a = 1 - h_a - e_a$. At smooth regions where $h_a = 0$ and $e_a = 0$, we have the slope:

$$\begin{aligned} k &= \Delta V_{a+1} - \Delta V_a \\ &= l_0 \Delta V_1 + h_0 \end{aligned} \tag{5}$$

2.1 solving MDP in bimodal

In bomal, the domain is $[0, d_2]$. Therefore, at age d_2, the modal has this property:

$$h_{d_2} + e_{d_2} = 1 \tag{6}$$

So d_2 is a critical point where:

$$\begin{aligned} V'_{d_2} &= h_{d_2} V_0 + h_{d_2} + e_{d_2} V_0 \\ V'_0 &= h_0(1 + V_0) + e_0 V_0 + l_0 V_1 \\ \Delta V_{d_2} &= V'_{d_2} - V'_0 \\ &= h_{d_2} - h_0 - l_0 \Delta V_1 \end{aligned} \tag{7}$$

The next critical point we look at is d_1, by computing ΔV_{d_1}'s value propagating from 0 and back propagating from d2, we could has another relation between slope k and ΔV_1.

$$\begin{aligned} \Delta V_{d_1} &= l_{d_1} \Delta V_{d_1+1} - k + h_{d_1} \\ &= \Delta V_1 + k(d_1 - 1) \end{aligned} \tag{8}$$

$$\Delta V_1 + (d_1 - l_{d_1} d_1 + l_{d_1} d_2)k = 1 - e_{d_1} \tag{9}$$

Combining equation 5 and equation 8, we solve k:

$$\begin{aligned} k &= \frac{l_0(1 - e_{d_1})}{1 + l_0[(h_{d_1} + e_{d_1})d_1 + l_{d_1} d_2]} \\ &\approx \frac{1 - e_{d_1}}{(h_{d_1} + e_{d_1})d_1 + l_{d_1} d_2} \end{aligned} \tag{10}$$

References

[1] Nathan Beckmann and Daniel Sanchez. Modeling cache performance beyond lru. In *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 225–236. IEEE, 2016.