Question → Which cache line should be evicted?

Hawkeye
- → Reconstructs belady's optimal for past
- → Learn OPT behaviour of past to inform eviction decisions for future loads

Two problems with OPT

- → efficient mechanism of reconstructing OPT → Liveliness intervals
- → Long history is needed to compute OPT → Set dueling

<u>Solutions</u>

Key points: → OPT decision depends on the overlap of reuse intervals
→ OPT decision for past access can be determined at next use
→ reuse of 8x is enough

Incoming line
- → cache-friendly ⇒ Inserted with high priority
- → cache-averse ⇒ Marked as eviction candidates

$$\begin{tikzcd}
& \text{Hawkeye} \arrow[dl] \arrow[dr] & \\
\text{OPTgen} \arrow[d] & & \text{Hawkeye predictor} \arrow[d] \\
\text{Simulates OPT behaviour} \arrow[d] & & \text{Eviction decision} \\
\text{Produce inputs to train} \arrow[uurr] & &
\end{tikzcd}$$

Belady's OPT:

→ Evict the farthest reuse block.

$$\begin{tikzcd}
\text{Cache replacement problem} \arrow[r, Rightarrow] & \text{Binary classification problem} \arrow[d] \arrow[dr] & \\
& \text{Cache friendly} & \text{Cache averse}
\end{tikzcd}$$

* In fact if a certain load instruct has brought in lines which caused hits, then there is a high prob the same instruction would bring in lines that would be hit.

Doubt: PC ? → Program counter?

→ Too much store overhead!

↓

Set Dueling!

$$\begin{tikzcd}
\text{Hawkeye} \arrow[r] \arrow[dr] \arrow[ddr] & \text{History} \arrow[r, Rightarrow] & \text{Sampled cache} \\
& \text{Decision} \arrow[r, Rightarrow] & \text{Occupancy vector \& OPTgen} \\
& \text{Training} \arrow[r, Rightarrow] & \text{PC based predictor}
\end{tikzcd}$$

$$\begin{tikzcd}
\text{Access} \arrow[d] & & \\
\text{Sampled cache} \arrow[r] & \text{If tag match} \arrow[r] & \text{occupancy vector} \arrow[d] \\
& & \text{Binary prediction} \arrow[d] \\
& \underline{\underline{\text{Based on the PC}}} \arrow[dr] & \text{Train predictor} \arrow[d] \\
& & \text{Assign RRPV value}
\end{tikzcd}$$

	Hit	Miss
Cache-averse	7	7
Cache-friendly	0	0
		age by 1 if all < 6.

Optgen and occupancy vector:

→ When a line is accessed put that as 0 in occupancy vector

→ Check all past, increment that to this by 1. If exceeds cache size, its a miss.

Doubt: Phase change boundary?

Evict cache averse

↓

No predicted cache averse,

↓

LRU

↓

evicts cache-averse of new working set.

Predictor detrained

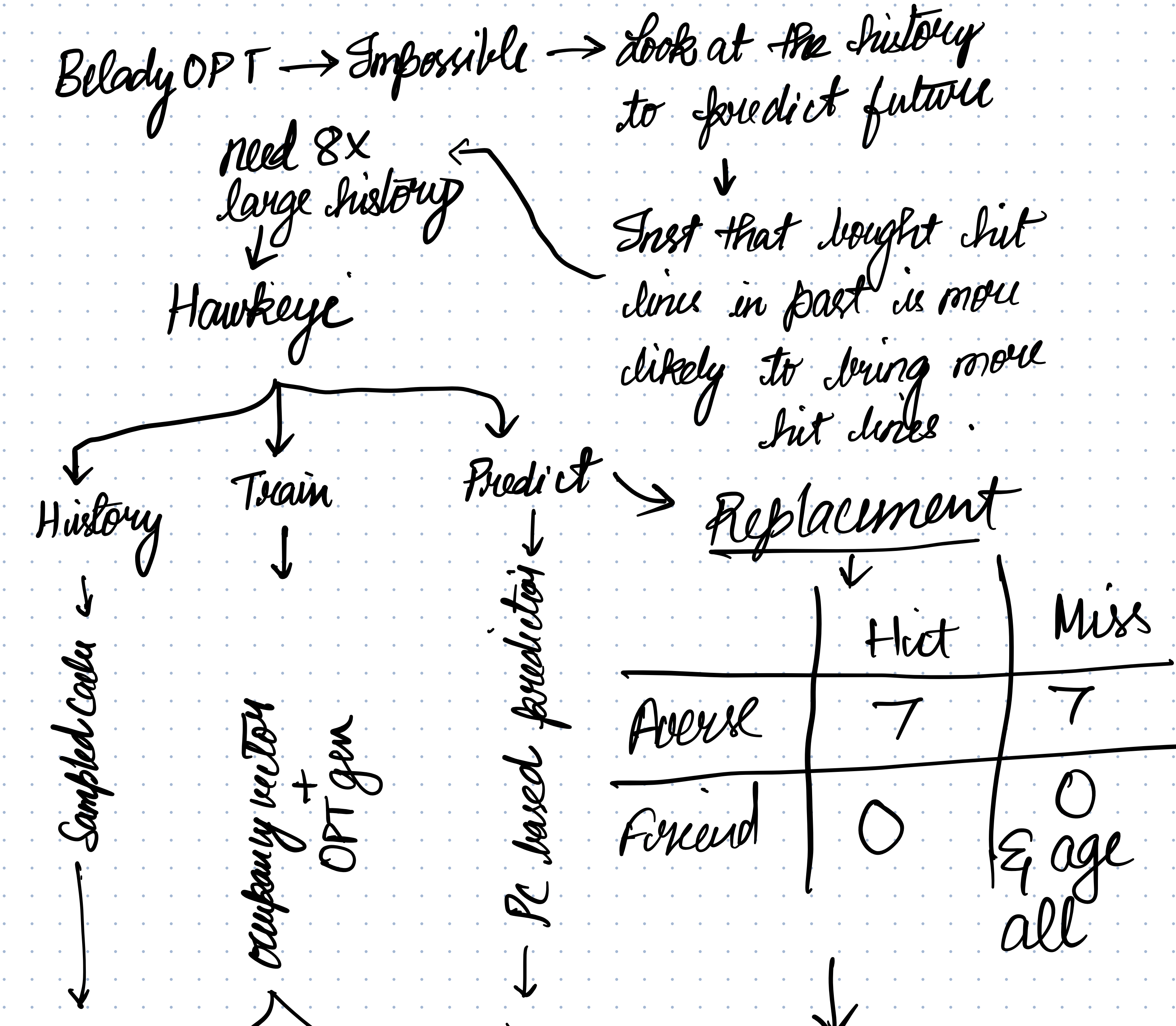
↓

when cache-friendly lines are evicted

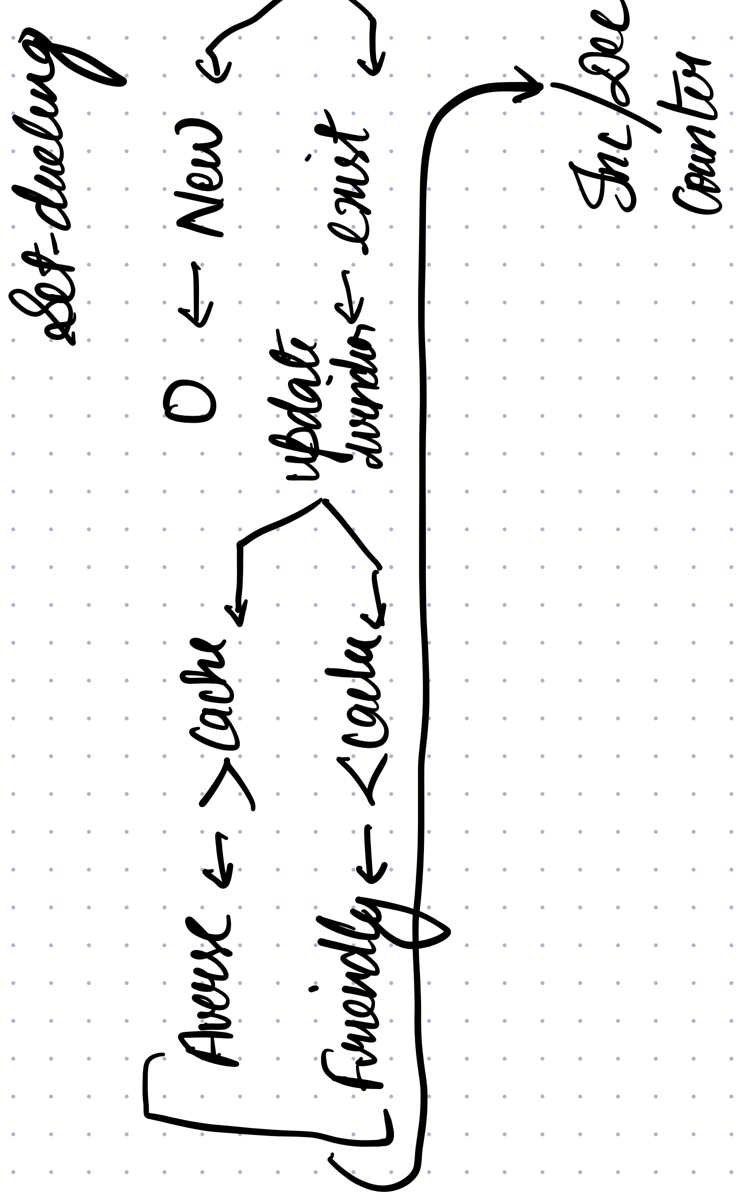
	Hit	Miss
averse	7	7
friendly	0	0

age
other
lines//

Mind map:


Belady OPT
Impossible
look at the history to predict future
Inst that bought hit lines in past is more likely to bring more hit lines.
need 8x large history
Hawkeye
History
Train
Predict
Replacement
Sampled cache
occupancy vector + OPT gen
PC based prediction

	Hit	Miss
Averse	7	7
Friend	0	0 & age all



No averse

↓

Indicates phase change

↓

LRU eviction

↓

detrain PC