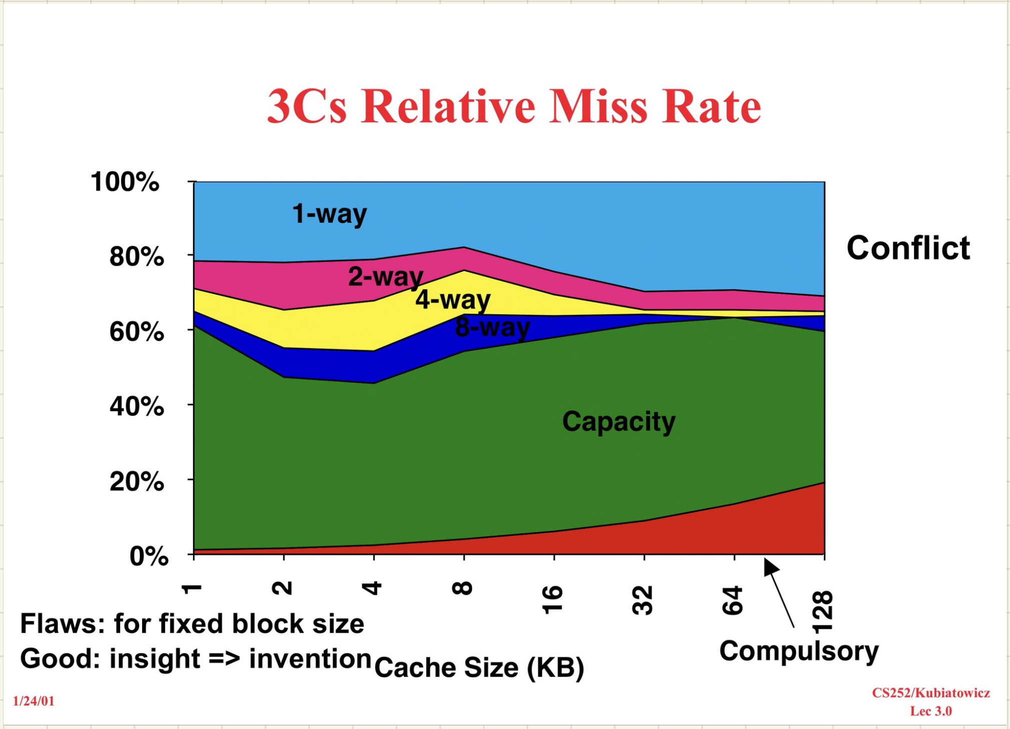
Review : Improving Cache Performance

1. Reduce the miss rate.

Classification : 3 "Cs"

Compulsory . Capacity . Conflict .

Recent : 4th "C" Coherence



3Cs Relative Miss Rate

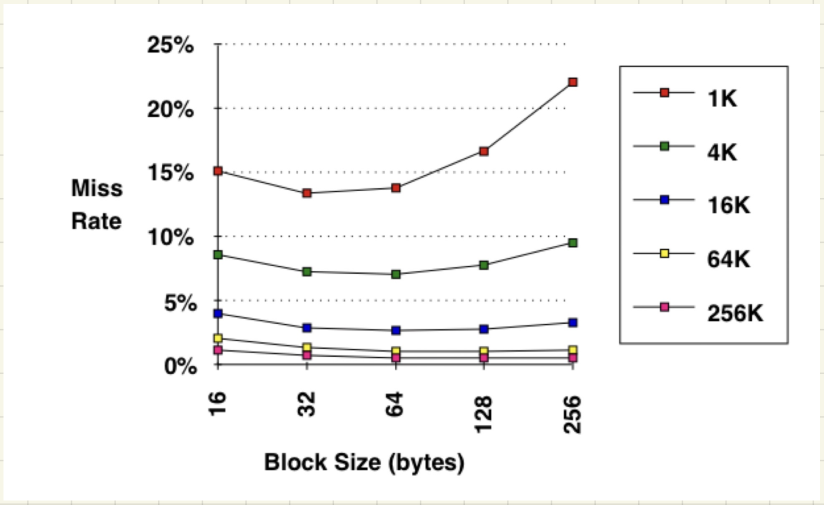
Flaws: for fixed block size

Good: insight => invention

1/24/01

CS252/Kubiatowicz Lec 3.0

Method (i) Reduce misses via larger block size

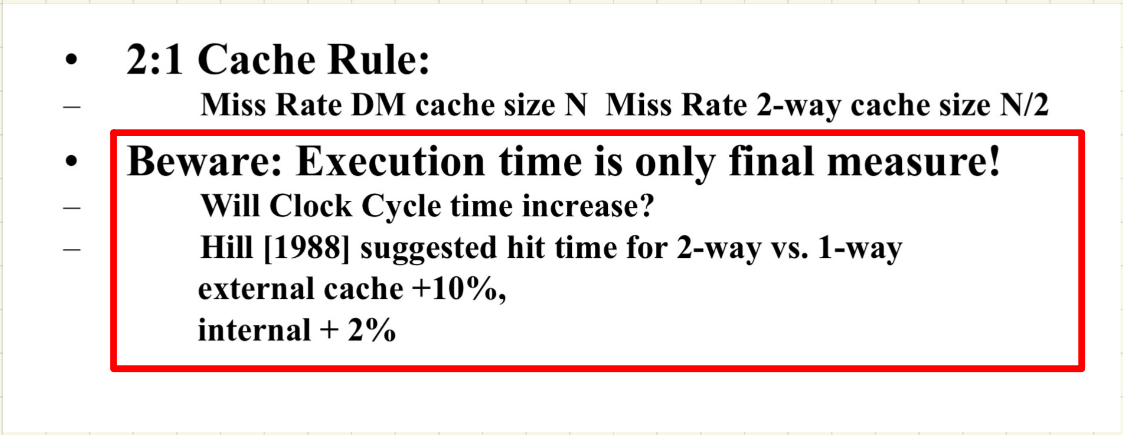


Miss rate rises after falls.

Firstly, larger blocks increase space locality. → fall

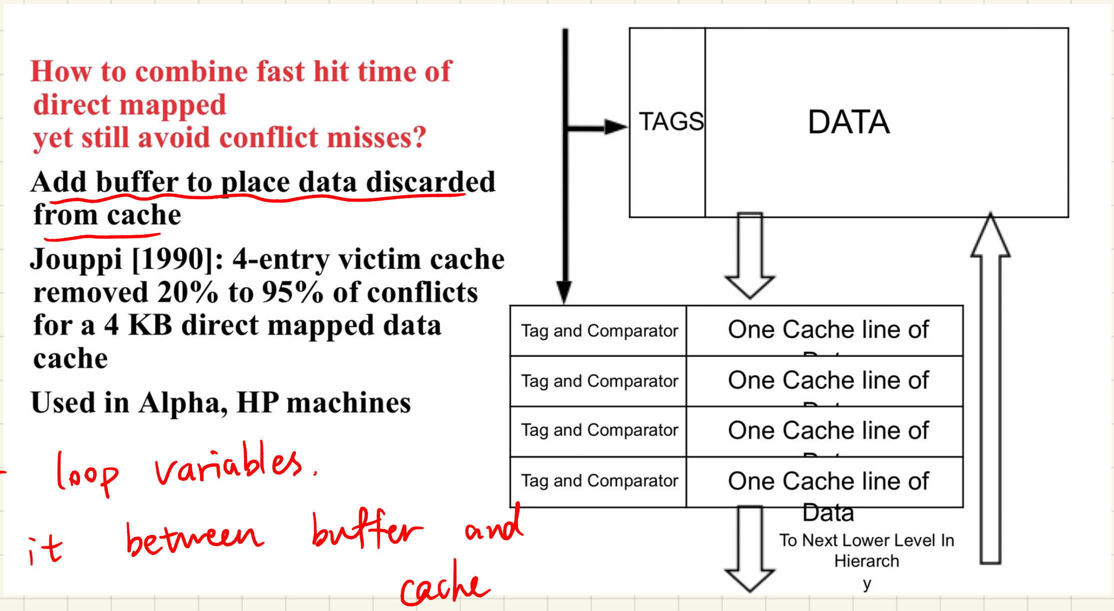
Secondly, larger blocks mean smaller block numbers, which causes more conflicts. And when the total size is small, this leads to capacity misses as well.

Method (ii) Reduce misses via higher associability



- **2:1 Cache Rule:**
  - Miss Rate DM cache size N Miss Rate 2-way cache size N/2
- **Beware: Execution time is only final measure!**
  - Will Clock Cycle time increase?
  - Hill [1988] suggested hit time for 2-way vs. 1-way external cache +10%, internal + 2%

Method (iii) Reduce misses via a "Victim Cache"



How to combine fast hit time of direct mapped yet still avoid conflict misses?

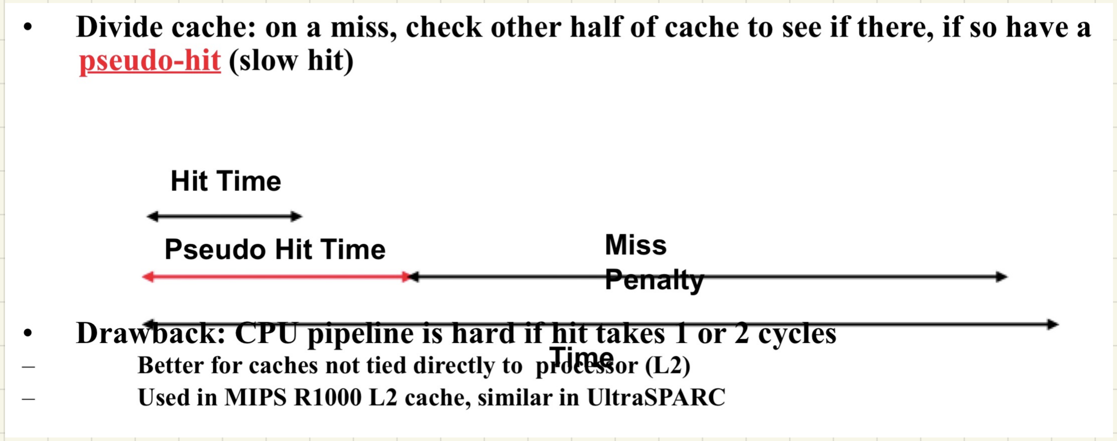
Add buffer to place data discarded from cache

Jouppi [1990]: 4-entry victim cache removed 20% to 95% of conflicts for a 4 KB direct mapped data cache

Used in Alpha, HP machines

consider loop variables, swap it between buffer and cache

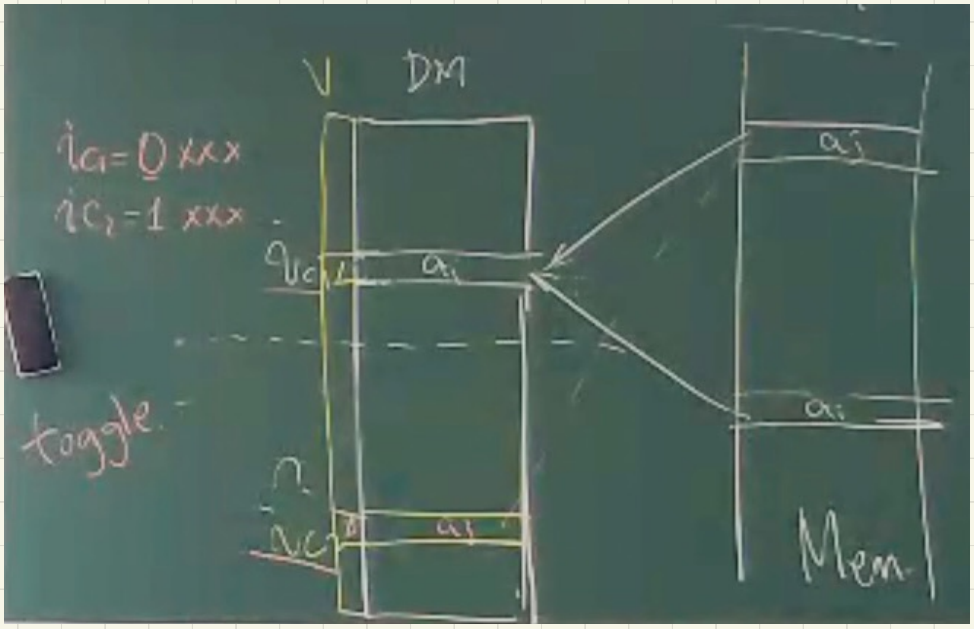
Method (iv) Reduce misses via "Pseudo-Associativity"



- Divide cache: on a miss, check other half of cache to see if there, if so have a pseudo-hit (slow hit)

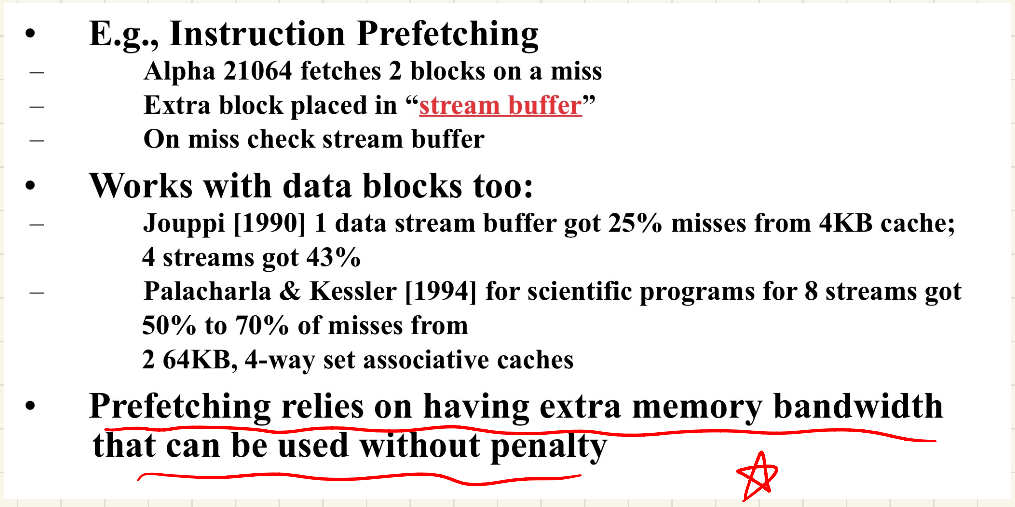
Hit Time
Pseudo Hit Time
Miss Penalty
Time

- Drawback: CPU pipeline is hard if hit takes 1 or 2 cycles
  - Better for caches not tied directly to processor (L2)
  - Used in MIPS R1000 L2 cache, similar in UltraSPARC



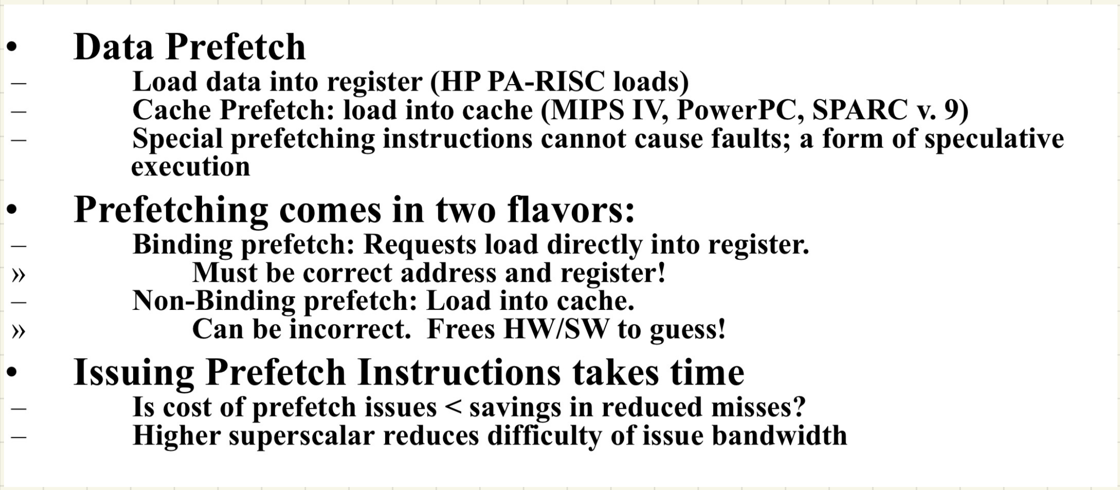
Method (v) Reduce misses by Hardware / Software Prefetching

Hardware :



- E.g., Instruction Prefetching
  - Alpha 21064 fetches 2 blocks on a miss
  - Extra block placed in "stream buffer"
  - On miss check stream buffer
- Works with data blocks too:
  - Jouppi [1990] 1 data stream buffer got 25% misses from 4KB cache; 4 streams got 43%
  - Palacharla & Kessler [1994] for scientific programs for 8 streams got 50% to 70% of misses from 2 64KB, 4-way set associative caches
- Prefetching relies on having extra memory bandwidth that can be used without penalty ☆

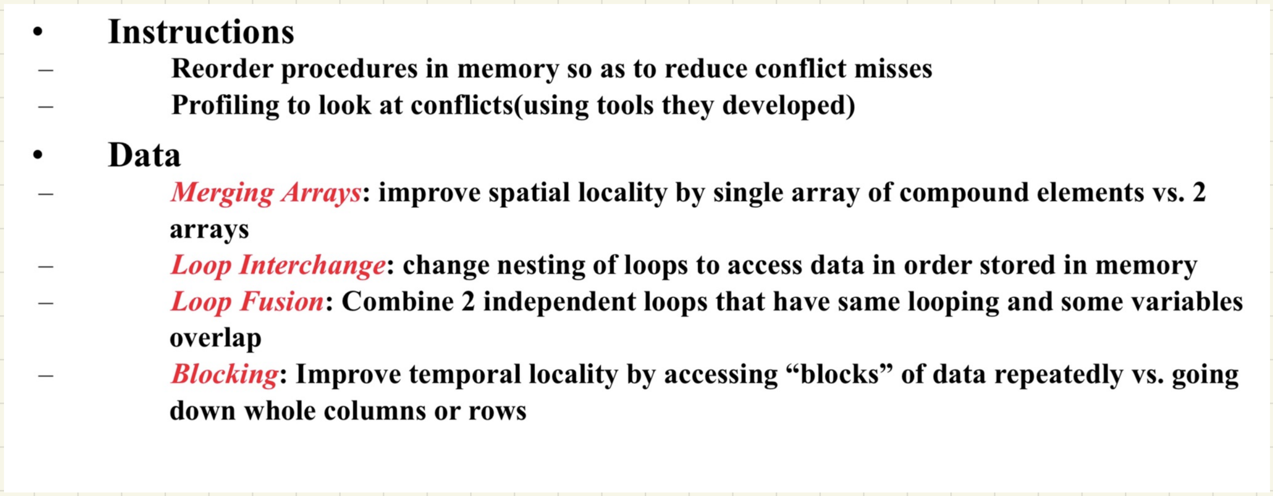
Software :



- **Data Prefetch**
  - **Load data into register (HP PA-RISC loads)**
  - **Cache Prefetch: load into cache (MIPS IV, PowerPC, SPARC v. 9)**
  - **Special prefetching instructions cannot cause faults; a form of speculative execution**
- **Prefetching comes in two flavors:**
  - **Binding prefetch: Requests load directly into register.**
    - **Must be correct address and register!**
  - **Non-Binding prefetch: Load into cache.**
    - **Can be incorrect. Frees HW/SW to guess!**
- **Issuing Prefetch Instructions takes time**
  - **Is cost of prefetch issues < savings in reduced misses?**
  - **Higher superscalar reduces difficulty of issue bandwidth**

☆ Method (vi) Reducing misses by Compiler Optimizations

Main idea:



- **Instructions**
  - **Reorder procedures in memory so as to reduce conflict misses**
  - **Profiling to look at conflicts(using tools they developed)**
- **Data**
  - ***Merging Arrays*: improve spatial locality by single array of compound elements vs. 2 arrays**
  - ***Loop Interchange*: change nesting of loops to access data in order stored in memory**
  - ***Loop Fusion*: Combine 2 independent loops that have same looping and some variables overlap**
  - ***Blocking*: Improve temporal locality by accessing "blocks" of data repeatedly vs. going down whole columns or rows**