

W6D1 Notes

章轩畅 2022.10.17

Outline

Question: the performance of Cache?

- 评价指标--AMAT(average memory access time)
 - $AMAT_{Mem} = T_{addressing} + T_{access} + T_{transfer}$, 其中 T_{access} 较大
 - $AMAT_{Cache} = T_{Hit} + \eta_{Miss} \cdot T_{MissPenalty}$
- Reason
- How to decrease AMAT ?
 - Miss-rate
 - Miss-penalty
 - Hit-time

Review

- Cache 产生原因: CPU与Memory发展差距大
- Cache 分类
 - Direct Mapped Cache: $N=1$
 - N-way Associative Cache--disadvantage: delay
 - Full Associative Cache: $N=\max$
- Cache performance

Impact on Performance

- **Suppose a processor executes at**
 - Clock Rate = 200 MHz (5 ns per cycle), Ideal (no misses) CPI = 1.1
 - 50% arith/logic, 30% ld/st, 20% control
- **Suppose that 10% of memory operations get 50 cycle miss penalty**
- **Suppose that 1% of instructions get same miss penalty**
- **CPI = ideal CPI + average stalls per instruction**
$$\begin{aligned} & 1.1(\text{cycles/ins}) + \\ & [0.30 (\text{DataMops/ins}) \\ & \quad \times 0.10 (\text{miss/DataMop}) \times 50 (\text{cycle/miss})] + \\ & [1 (\text{InstMop/ins}) \\ & \quad \times 0.01 (\text{miss/InstMop}) \times 50 (\text{cycle/miss})] \\ & = (1.1 + 1.5 + .5) \text{ cycle/ins} = 3.1 \end{aligned}$$
- **58% of the time the proc is stalled waiting for memory!**
- **AMAT=(1/1.3)×[1+0.01×50]+(0.3/1.3)×[1+0.1×50]=2.54**

----an example of calculation

Explanation:

- ideal CPI=1 cycle/ins
- Suppose 100 instr/program, 130次Mem访问=100次取指令+30次ld/st指令

- 由数据发现, miss-rate: ins<<mem, 于是有了指令和内存放在一起还是分开放的争论
Von Neumann(放在一起) V.S. Harvard(分开放)
- What happens on a write?
 - Write back: 数据改变后, 在Cache中被替换时写回内存
 - Write through: 数据改变后立即写回内存

Reducing Misses

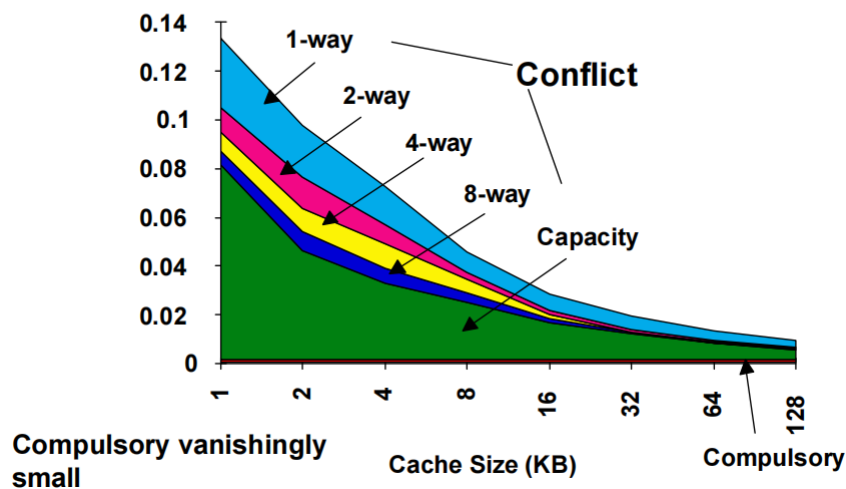
Miss 种类及大致解决思路

- compulsory--开始运行时的强制写入问题
解决思路: 提前预热, 读入预测的访问对象数据
- capacity--容量问题
解决思路: 扩容
- Conflict--取模后冲突问题

图像分析 (理解几种miss的原因)

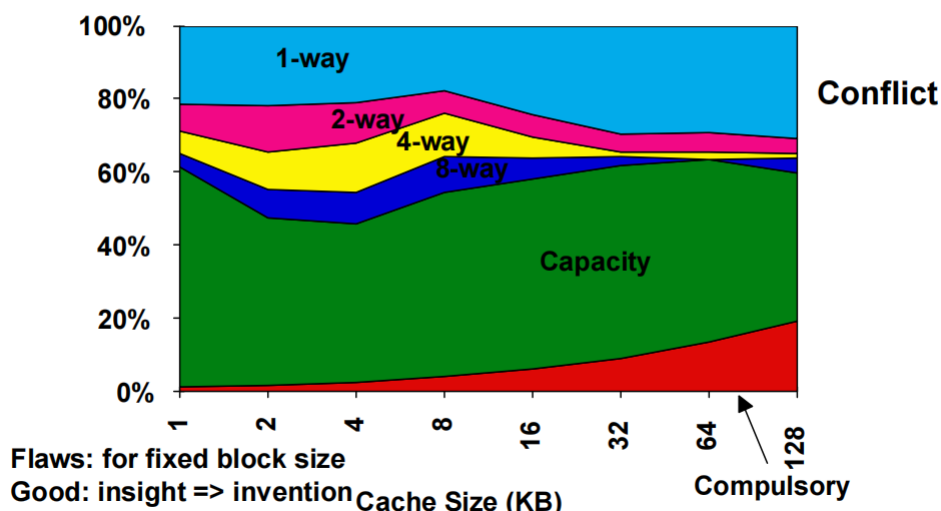
•

3Cs Absolute Miss Rate (SPEC92)



- way增加, 关联度增加, miss-rate减小--解决了Conflict问题
 - Cache size很大时, miss-rate随way增加而减小不明显--Cache size很大时, Conflict 问题较小
 - 工业上, 一般而言, 一级Cache(靠近core)采用直接映射, 中间的Cache采用多路映射, 提高相关性
-

3Cs Relative Miss Rate



图像呈现两头小中间大

- 右侧小，中间大与上图解释相同
- 左侧小--block过少（极端情况，只有1个）时，不论Cache种类，几乎必然miss

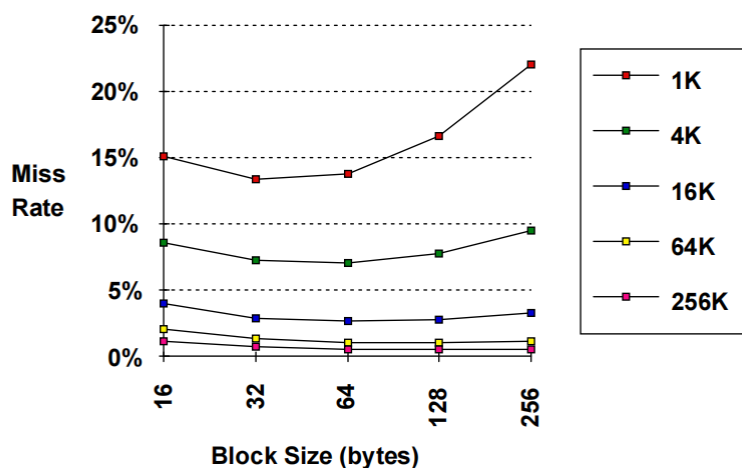
工业上大的缓存不太关心路数，多采用2路

- Cache size 很大，2路及以上miss-rate变化不大，路数增加还会导致工艺复杂的多路比较器

具体解决方法

Change Block Size

1. Reduce Misses via Larger Block Size



- 耦合线形状：两头低，中间高
 - 一开始下降--解决capacity问题
 - 后来上升--由于总容量固定，block size过大会导致行数过小，进而导致conflict问题
- 总容量大时，变化不明显
 - 相当于之前图线的局部放大

Example: Avg. Memory Access Time vs. Miss Rate

- Example: assume CCT = 1.10 for 2-way, 1.12 for 4-way, 1.14 for 8-way vs. CCT direct mapped

| Cache Size (KB) | Associativity | | | |
|--------------------|---------------|-------|-------|-------|
| | 1-way | 2-way | 4-way | 8-way |
| 1 | 2.33 | 2.15 | 2.07 | 2.01 |
| 2 | 1.98 | 1.86 | 1.76 | 1.68 |
| 4 | 1.72 | 1.67 | 1.61 | 1.53 |
| 8 | 1.46 | 1.48 | 1.47 | 1.43 |
| 16 | 1.29 | 1.32 | 1.32 | 1.32 |
| 32 | 1.20 | 1.24 | 1.25 | 1.27 |
| 64 | 1.14 | 1.20 | 1.21 | 1.23 |
| 128 | 1.10 | 1.17 | 1.18 | 1.20 |

(Red means A.M.A.T. not improved by more associativity)

- 实际效果不能看miss-rate, 要看AMAT
- 黑色数据--随着way的增加, miss-rate减少, AMAT进而减少
- 红色数据反常--随着way的增加, 虽然miss-rate减少了, 但是hit-time增加了, 当hit-time影响更大时, AMAT增加

Change Associativity

Change Compiler