- architecture and OS
- Memory
  - properties
    - caching
      - cache accessing address
        -
      - accessing process
        - first find "set" in cache, then find "target"
      - accessing state
        - hit
          - block find in cache with corresponding set and same target
            - the 'target' means the block position in main memory
        - miss
          - compulsory (cold) miss
            - occur because cache starts empty
            - first reference to the block
          - capacity miss
            - when the set of active cache blocks is larger than the cache
            - ?
          - conflict miss
            - the n-th reference to the block (n>1), and the block is not in the cache
      - cache organization (S, E, B)
        -
        - extension
          - introduce a "dirty" bit in block to increase efficiency
          - only write back when the the cache line is going to be replaced
    - performance metrics
      - Miss Rate
        - Fraction of memory accesses not found in cache (misses / accesses) = 1 - hit rate
        - Typical numbers (as %)
          - 3-10% for L1
          - can be quite small (e.g., < 1%) for L2, depending on size, etc.
      - Hit Time
        - Time to deliver a cached block to the processor
          - includes time to determine whether line is in cache
        - Typical numbers
          - 4 clock cycle for L1
          - 10 clock cycles for L2
      - Miss Penalty
        - Additional time required because of a miss
        - typically 50-200 cycles for main memory (Trend: increasing)
  - locality
    - types
      - Temporal locality
        - Recently referenced items are likely to be referenced again in the near future
        - same address
      - Spatial locality
        - Items with nearby addresses tend to be referenced close together in time
        - nearby address
    - importance
      - principle
        - Programs tend to use data and instructions with addresses near or equal to those they have used recently
  - memory mountain
    - origin
      -
    - definition
      -
      - Closer Look at Stride Effects
        - Working set size (log2)
        - Stride (words)
        - Read throughput (MB/s)
        - Ridges of Temporal Locality
        - Slopes of Spatial Locality
        - Aggressive prefetching
        - Multiple strides through memory
        - Cache blocks
        - Cache size
        - 32 KB L1
        - 256 KB L2
        - 8 MB L3
        - Main memory
        - Mtn. Hill: read throughput vs. stride for working sets that fit in L1 cache
        - Mtn. Valley: read throughput vs. working set size for stride = 1 word
        - Mtn. Ridge: read throughput vs. stride for working sets that fit in L2 cache
        - Mtn. Plain: read throughput vs. stride for working sets that fit in L3 cache
        - Mtn. Foothills: read throughput vs. stride for working sets that do not fit in cache
        - Mtn. Ice: read throughput vs. stride for working sets that do not fit in cache
        - 6.9GB/sec
        - 2.9GB/sec
        - 1.9GB/sec
        - 1.3GB/sec
        - 900MB/sec
        - 600MB/sec
        - 350MB/sec
    - lesson
      - Rearranging loops to improve spatial locality
        - e.g. matrix multiplication
          -
          -
          -
          - performance virtualization
            -
      - Using blocking to improve temporal locality
        - e.g. matrix multiplication
          - normal
            -
          - blocking
            -
            - a^n double array
            - array blocking size = L * L
            - cache block size = 8 double
            - cache could afford storing more than 2 array block
            - so when inverting each pair of array block, it need at most 2 * L^2 / cache block size cache misses, and there are n^2/L^2 pairs in this pair of array block in the cache
            - and we need n/L iteration
            - so the overall misses is (n/L)^2 (2 * L^2 / B) = n(L+B, which is far more smaller than the cost of normal case
          - further optimization
            - avoid disposal eviction
              - Prerequisites
                - elements in same position in different arrays will be stored in the same set in cache
              - for example, when we are handling c[x][y] = a[i][k] * b[k][i]
              - first considering c[0][0] + a[0][0] * b[0][1], we put b[0] into set 0, a[1] into set 1,
              - then considering c[1][1] + a[1][1] * b[1][1], we first access a[1], then we evict a[1] and put b[1] into set 1,
              - attention that in the following operations, the a[1] is required again, so we need to evict b[1] and put back a[1], here comes the extra cost,
              - so what we can do is to delay the assignment of the diagnal situation, by how we could cut the cost
  - Virtual Memory
- Storage Tech
  - magnetic medium
  - Disk
    - componet
      - each surface consists of cocentric rings - tracks
        -
      - each track consists of sectors separated by gaps
        -
    - service time
      - $T_{access} = T_{seek time} + T_{rotation} + T_{transfer}$
      - disk access time
        - seek time (T avg seek)
          - Time to position heads over cylinder containing target sector.
          - a "fixed" number
          - Typical $T_{avg\ seek}$ is 9 ~ 6 ms
        - rotational latency (T avg rotation)
          - Time waiting for first bit of target sector to pass under r/w head.
          - $T_{avg\ rotation}$ = 1/2 x 1/RPMs x 60 sec/1 min
          - Typical rotational rate = 7,200 RPMs
          - per min
        - transfer time (T avg transfer)
          - Time to read the bits in the target sector.
          - $T_{avg\ transfer}$ = 1/RPMs * (avg #sectors/track) * (60 sec/1 min)
      - e.g.
        - Disk Access Time Example
          - Given:
            - Rotational rate = 7,200 RPM
            - Average seek time = 9 ms.
            - Avg # sectors/track = 400.
          - Derived:
            - $T_{avg\ rotation}$ = 1/2 x (60 sec/7200 RPM) x 1000 ms/sec = 4 ms.
            - $T_{avg\ transfer}$ = 60/7200 RPM x 1000 ms/sec x 1/400 = 0.02 ms
          - Important points:
            - Access time dominated by seek time and rotational latency.
            - First bit in a sector is the most time-consuming to access.
            - SRAM access time is about 4 ns/doubleword, DRAM about 60 ns
            - Disk is about 40,000 times slower than SRAM
            - 2,500 times slower than DRAM.
    - read disk
      - 1 CPU initiates a disk read by writing a command, logical block number, and destination memory address to a port (address) associated with disk controller.
      - 2 Disk controller reads the sector and performs a direct memory access (DMA) transfer into main memory.
      - 3 When the DMA transfer completes, the disk controller notifies the CPU with an interrupt (i.e., asserts a special "interrupt" pin on the CPU).
  - Nonvolatile Memories
    - retain information when powered off
    - types
      - Read-only memory (ROM) ——— programmed during production
      - Electrically erasable PROM (EEPROM) ——— electronic erase capability
      - Flash memory
        - EEPROMs, with partial (block-level) erase capability
        - Wears out after about 100,000 erasings
      - 3D XPoint (Intel Optane) & emerging NVMs ——— New materials
    - uses
      - Firmware programs stored in a ROM (BIOS, controllers for disks, network cards, graphics accelerators, security subsystems,...)
      - solid state disks (replacing rotating disks)
        - advantage
          - No moving parts → faster, less power, more rugged
        - disadvantage
          - Have the potential to wear out
          - In 2022, about 2 times more expensive per byte
        - application
          - Smartphones, laptops
          - Increasingly common in desktops and servers
      - Disk caches
  - accessing speed of different devices
    -

Presented with xmind