Microarchitecture

~ 34% of instructions access data memory

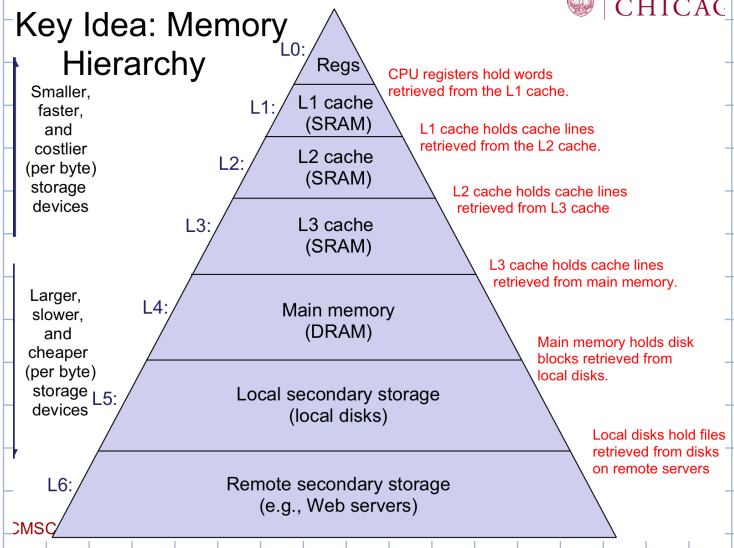
100% instructions stored in instruction memory

DRAM -> memory main also SRAM

accessing memory is slow at

why slow? technology

Key Idea: Memory Hierarchy

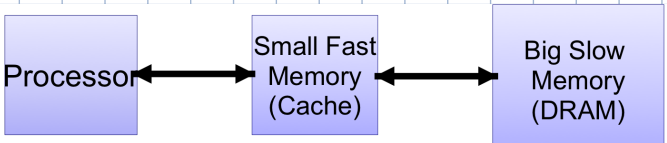


cache

DRAM

local storage

remote storage



Capacity: Register << Cache << DRAM

Speed: Register >> Cache >> DRAM

if data in fast memory -> low latency (cache or register)

why hierarchy? ~ speed

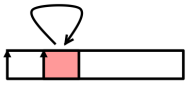
Locality

bridge fast CPU & slow memory

Principle of Locality: Programs tend to use data and instructions with addresses near or equal to those they have used recently

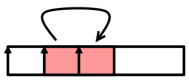
Temporal locality:

Recently referenced items are likely to be referenced again in the near future



Spatial locality:

Items with nearby addresses tend to be referenced close together in time



```
sum = 0;
for (i = 0; i < n; i++)
    sum += a[i];
return sum;
```

Data references

Reference array elements in succession (stride-1 reference pattern). — Spatial locality — next element is next address

Reference variable sum each iteration. — Temporal locality — being used over & over again

Instruction references

Reference instructions in sequence. — Spatial locality — close references

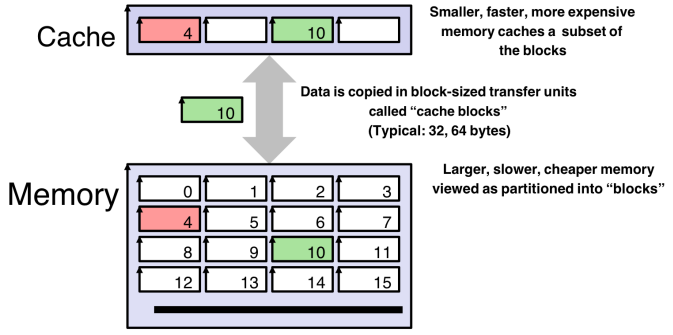
Cycle through loop repeatedly. — Temporal locality — referenced constantly again & again

lower memory → slower

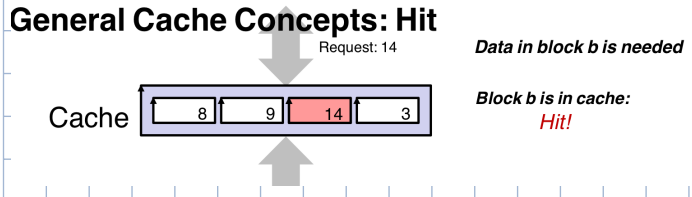
cache → fundamental idea of memory hierarchy. Smaller, faster acts like staging for subset of data in larger memory

General Cache Concepts

24:00



General Cache Concepts: Hit



hit rate = hits / accesses

want higher hit rate

miss: have to go to lower memory then return

misses / accesses