Caches

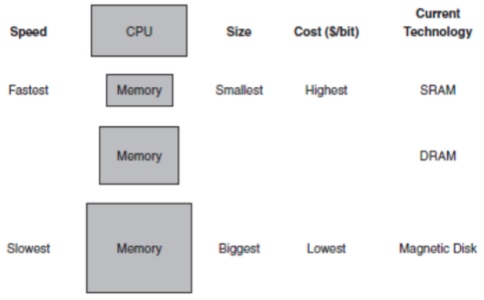
In an ideal computing system there would be unlimited fast memory access. However such a system is not practically achievable. Instead a memory heirarchy provides the illusion of large amounts of fast memory by organizing different types of storage to optimize speed and access

The principal of locality refers to the observation that programs tend to access only a small portion of their address space at any given time, which is crucial to optimizing the memory hierarchy. There are 2 main types of locality:

1. Temporal locality: Refers to the tendency of programs to access the same memory locations repeatedly within a short period. For example, instructions in a loop and induction variables.
2. Spacial locality: Denotes the likelihood of accessing memory locations close to those recently accessed. This occurs commonly in sequential instruction access and accessing array data.

Understanding these principals helps in designing cache systems and hierarchies to enhance performance.

Memory Hierarchy Levels



- cache levels include smaller faster memory that is closer to the processor.
- Data moves through layers of memory to improve speed and efficiency. Frequently accessed and nearby items are copied from permanent storage (like SSD or disk) to main memory (DRAM), and then to smaller, faster cache (SRAM) near the CPU.
- Data is moved between levels in blocks, which may include multiple words. This is a key concept in managing memory transitions.

Caches are used to store frequently accessed data closer to the processor, minimizing access times and improving speed.

Cache memory is the closest level of memory hierarchy to the CPU. It serves to swiftly retreive the most frequently or recently accessed data.

A hit occurs when accessed data is present in the upper memory level, leading to a quick data fetch. The hit ratio is a metric that denotes the number of hits divided by total accesses

A miss occurs when accessed data is absent in the cache, requiring a fetch