

08-B1

高级搜索树

B-树：大数据

640K ought to be enough for anybody.

- B. Gates, 1981

白嘉轩一听就不由得火了：“又是个百日忌讳！”仙草却说：“百日又不是百年。你权当百日后才娶我。你就忍一忍，一百天很快就过去了……”

邓俊辉

deng@tsinghua.edu.cn

现实A: 存储器容量的增长速度 << 应用问题规模的增长速度

1 Kilobyte = $2^{10} = 10^3$

1 Megabyte = $2^{20} = 10^6$

1 Gigabyte = $2^{30} = 10^9$

1 Terabyte = $2^{40} = 10^{12}$

1 Petabyte = $2^{50} = 10^{15}$

1 Exabyte = $2^{60} = 10^{18}$

1 Zettabyte = $2^{70} = 10^{21}$

1 Yottabyte = $2^{80} = 10^{24}$

1 Nonabyte = $2^{90} = 10^{27}$

1 Doggabyte = $2^{100} = 10^{30}$

❖ 存储器? - RAM: 不就是...无限可数个...寄存器吗!

- Turing: 不就是...无限长的...纸带吗!

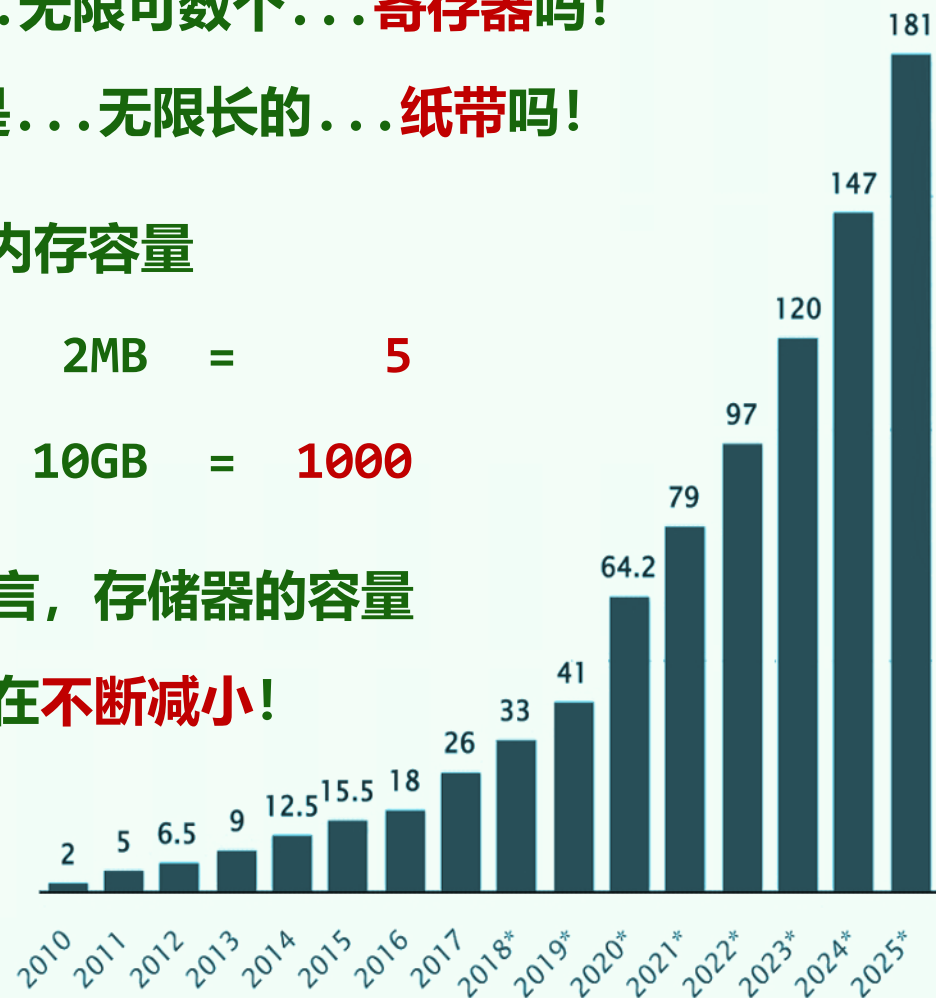
❖ 典型的数据库规模/内存容量

1990: 10MB / 2MB = 5

2020: 10TB / 10GB = 1000

❖ 相对而言, 存储器的容量

实际上在不断减小!



现实B：在特定工艺及成本下，存储器都是容量与速度的折中产物

1 Kilobyte = $2^{10} = 10^3$

1 Megabyte = $2^{20} = 10^6$

1 Gigabyte = $2^{30} = 10^9$

1 Terabyte = $2^{40} = 10^{12}$

1 Petabyte = $2^{50} = 10^{15}$

1 Exabyte = $2^{60} = 10^{18}$

1 Zettabyte = $2^{70} = 10^{21}$

1 Yottabyte = $2^{80} = 10^{24}$

1 Nonabyte = $2^{90} = 10^{27}$

1 Doggabyte = $2^{100} = 10^{30}$

❖ 为何...不把存储器做得...更大些？非不为，实不能！

❖ 存储器越大、越快，成本也越高

❖ 存储器容量越大/小，访问速度越慢/快



现实C：实用的存储系统，由不同类型的存储器级联而成，以综合其各自的优势

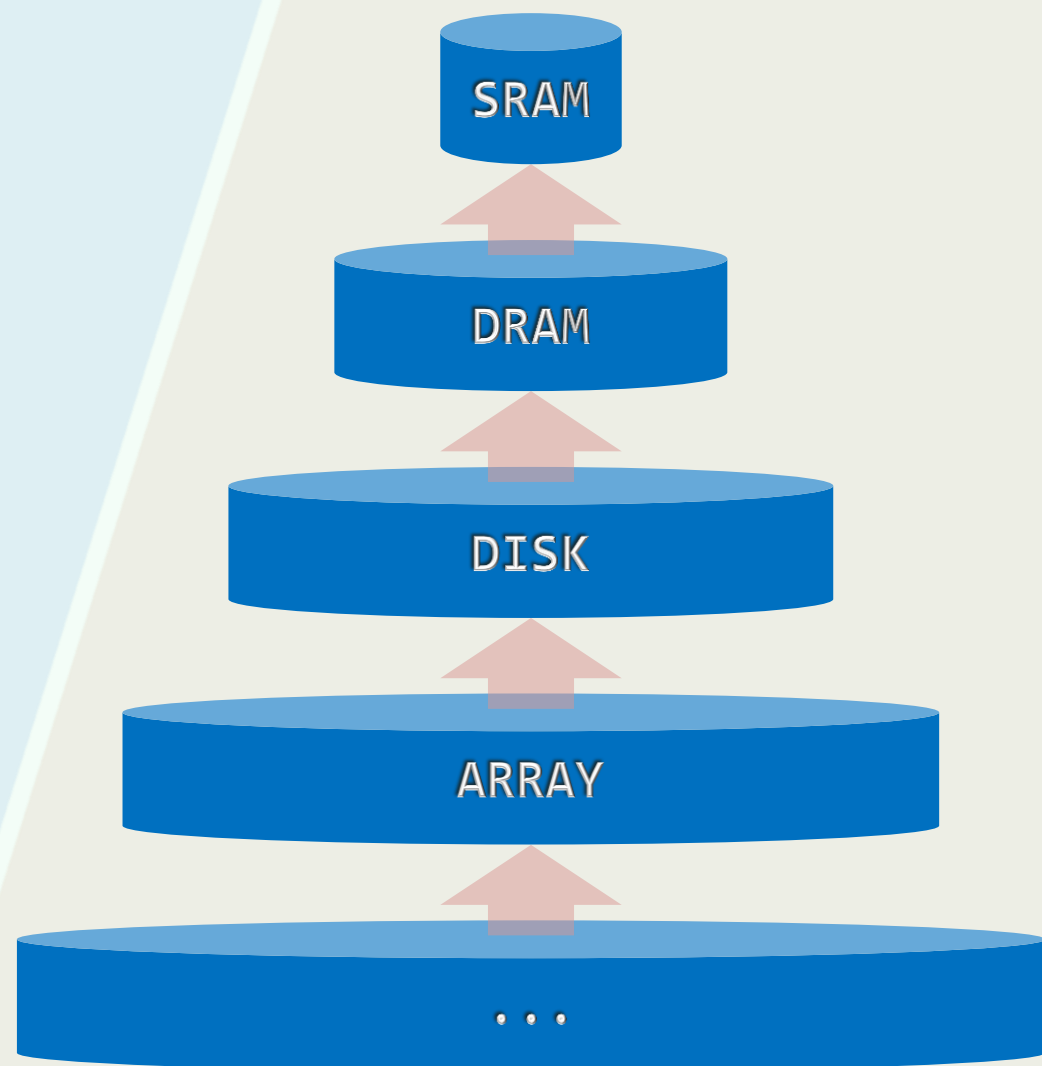
❖ 不同类型的存储器，容量、访问速度差异悬殊

	#cycles	sec
CPU Register :	0	ns
SRAM/cache :	4~75	ns
DRAM/main memory :	10^2	ns
DISK :	10^7	ms

❖ 若一次内存访问需要一秒，则一次磁盘访问就需一天

为避免一次磁盘访问，我们宁愿访问内存1000次

❖ 在分级的存储系统中，各类存储器有其各自的角色



分级存储：利用数据访问的局部性

❖ 机制与策略

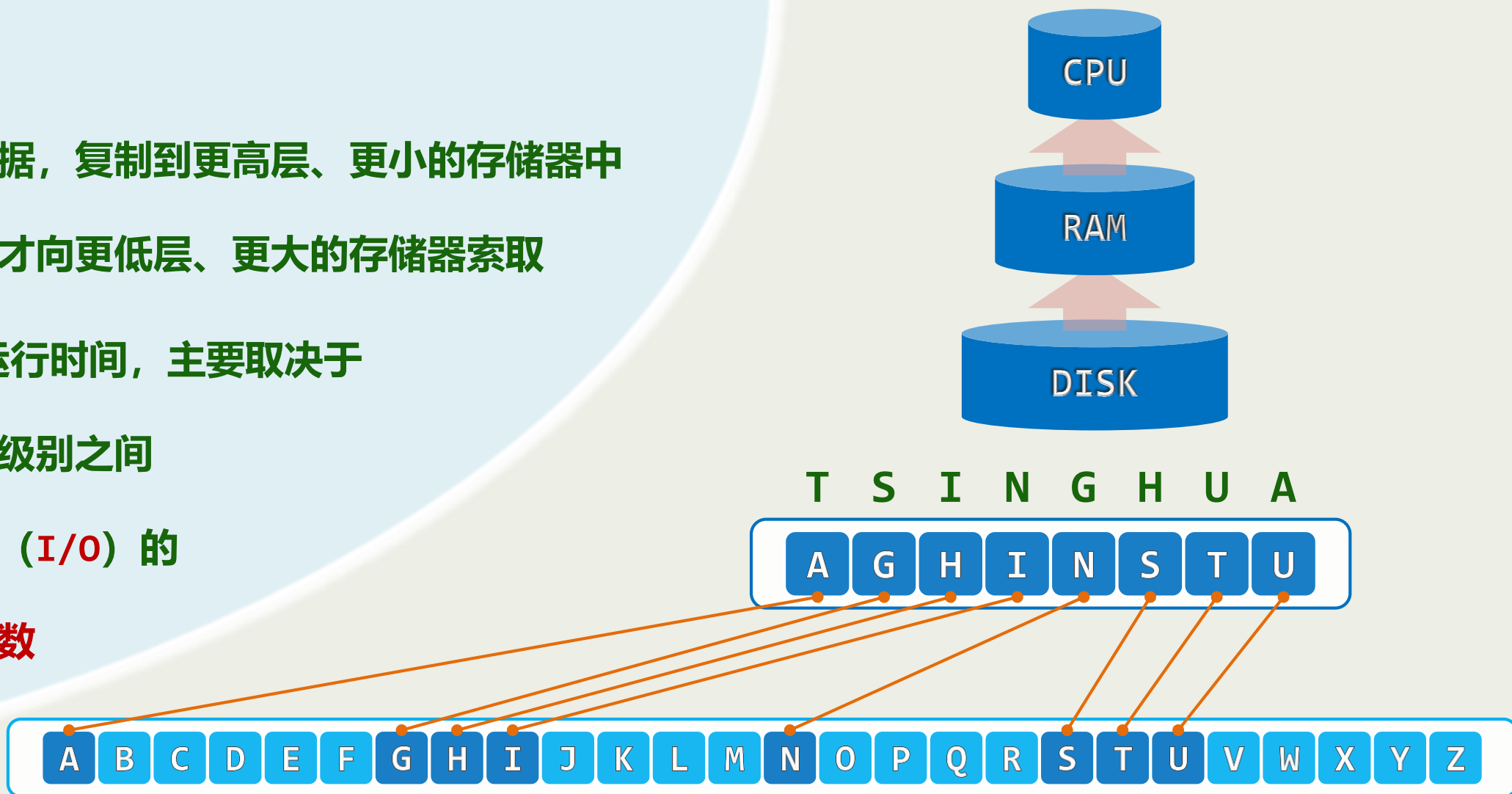
- 常用的数据，复制到更高层、更小的存储器中
- 找不到，才向更低层、更大的存储器索取

❖ 算法的实际运行时间，主要取决于

相邻存储级别之间

数据传输 (I/O) 的

速度与次数



分级存储：批量访问：在外存读写1B，与读写1KB几乎一样快

❖ 以页（page）或块（block）为单位，借助缓冲区，可大大缩短单位字节的平均访问时间

❖ #include <stdio.h>

```
#define BUFSIZ 512 //缓冲区默认容量
```

```
int setvbuf( //定制缓冲区
```

```
FILE* fp, //流
```

```
char* buf, //缓冲区
```

```
int _Mode, //_IOFBF | _IOLBF | _IONBF
```

```
size_t size); //缓冲区容量
```

```
int fflush( FILE* fp ); //强制清空缓冲区
```

