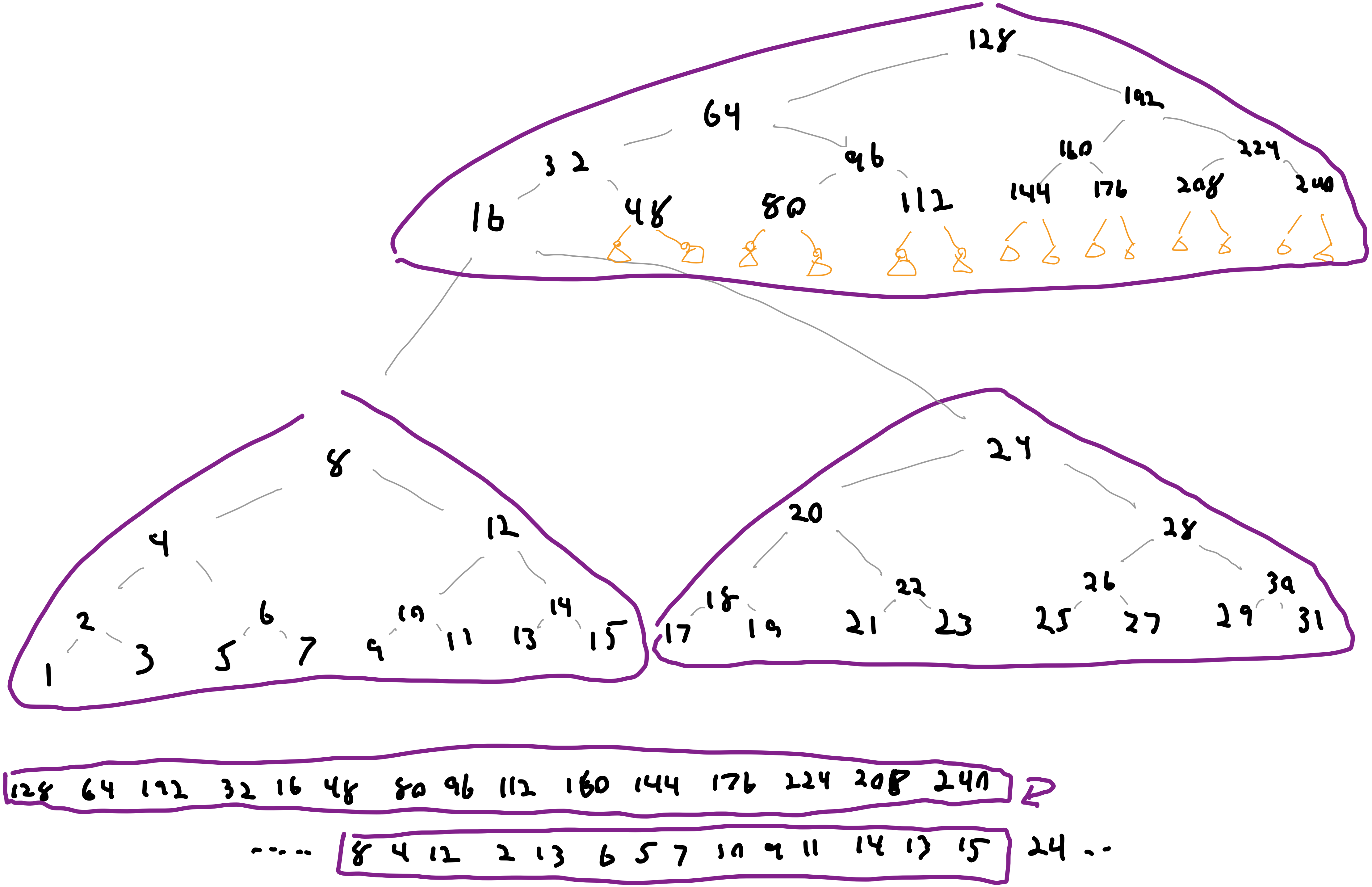


128 64 192 32 16 48 80 96 112 160 144 176 224 208 240
..... 8 4 12 2 1 3 6 5 7 10 9 11 14 13 15 24 ..

Algoritms

Model of Computation

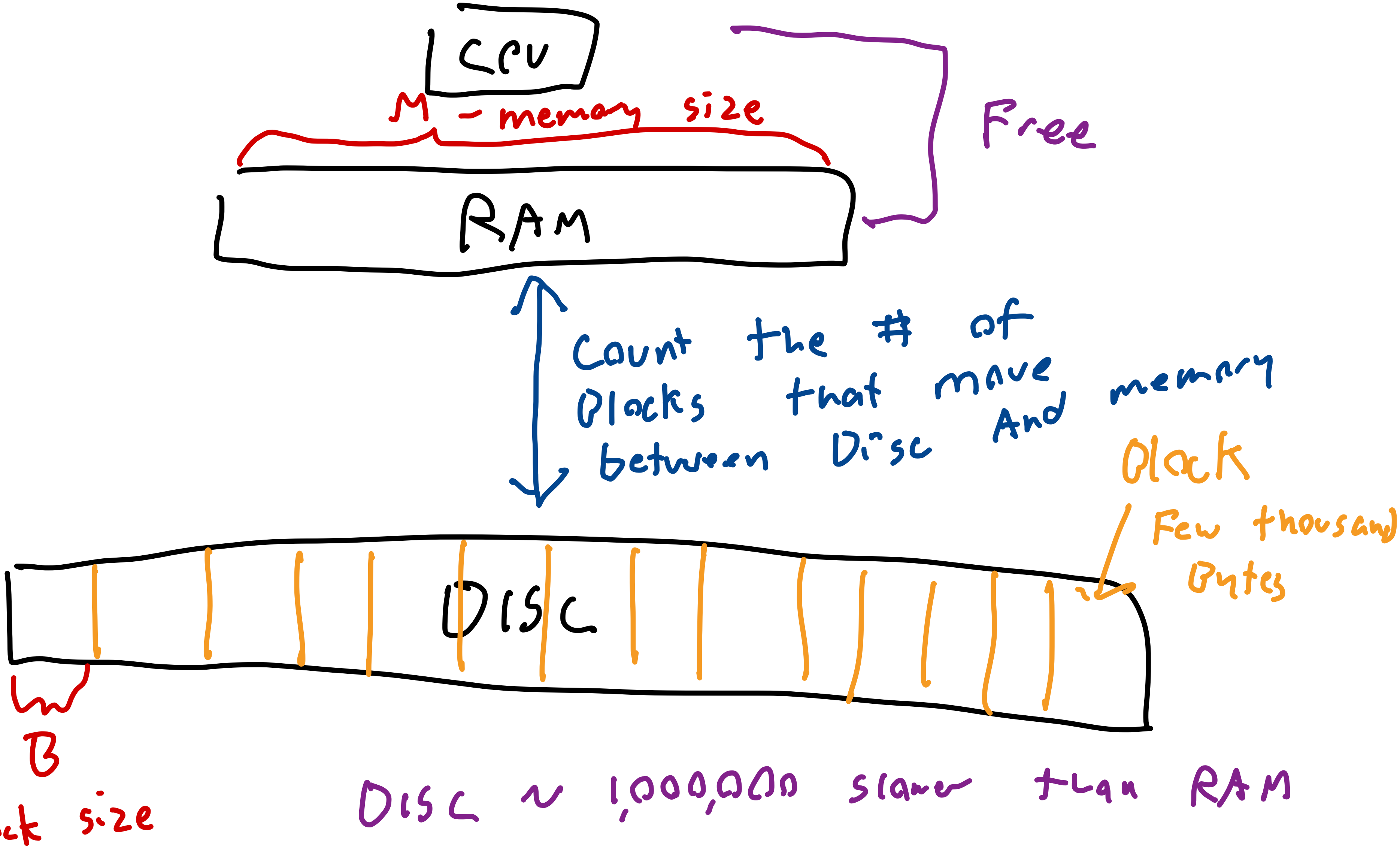
Computers

RAM Model / Standard

"Count the nuber of operations"

# External-Memory Model

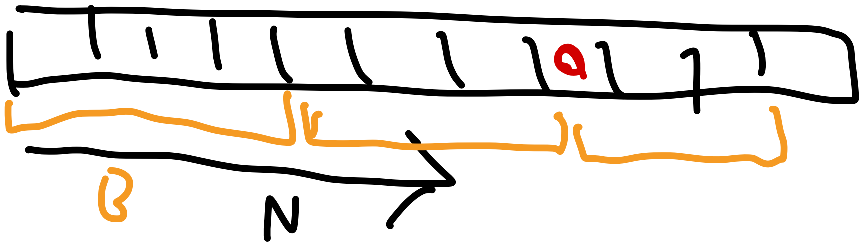
$\boxed{\text{D}}$isk
$\boxed{\text{A}}$ccess
$\boxed{\text{M}}$odel

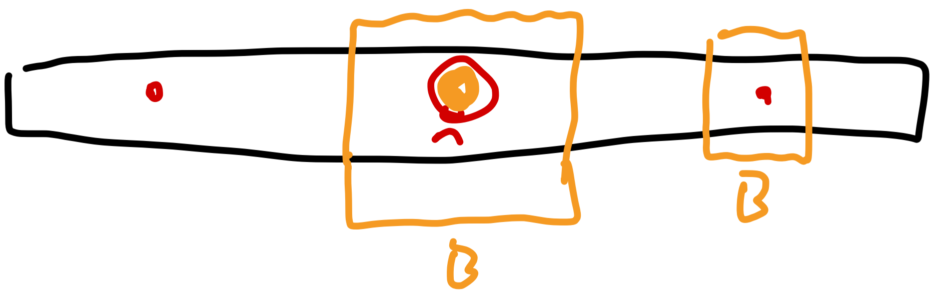



Disc ~ 1,000,000 slower than RAM

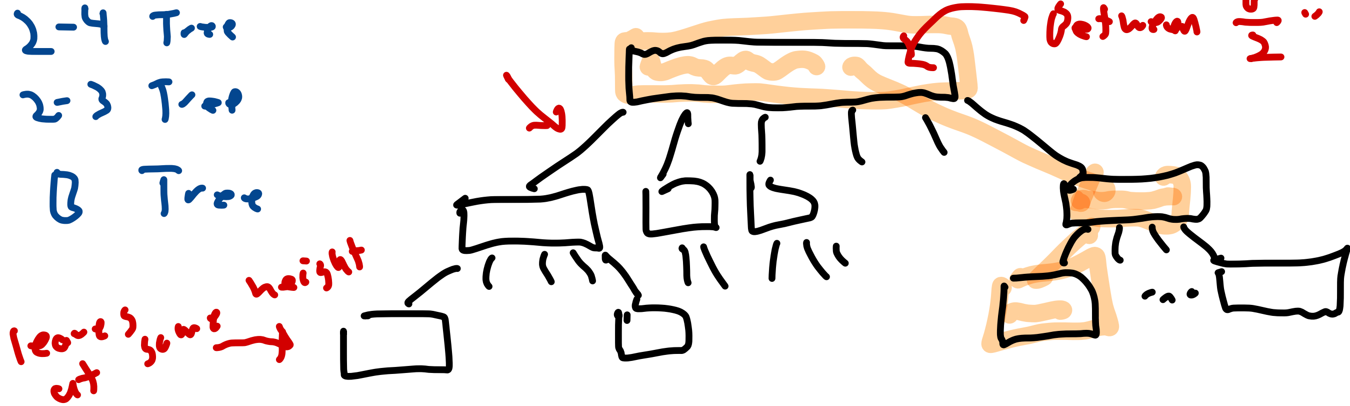
# Searching

| | | RAM | DAM |
|---|---|---|---|
| Linear Search | Search | $O(N)$ | $O(\frac{N}{B})$ |
| | Ins/Del | $O(N)$ | $O(\frac{N}{B})$ |
| Binary Search | Search | $\boxed{O(\log_2 N)}$ | $O(\log_2 N)$ |
| | Ins/Del | $O(N)$ | $O(\frac{N}{B})$ |
| 2-4 Tree, 2-3 Tree, B Tree | Search | $O(B \log_B N)$, $O(\frac{B}{\log B} \log_2 N)$ | $\boxed{O(\log_B N)}$ |
| | Ins/Del | $O(\frac{B}{\log B} \log_2 N)$ | $O(\log_B N)$ |



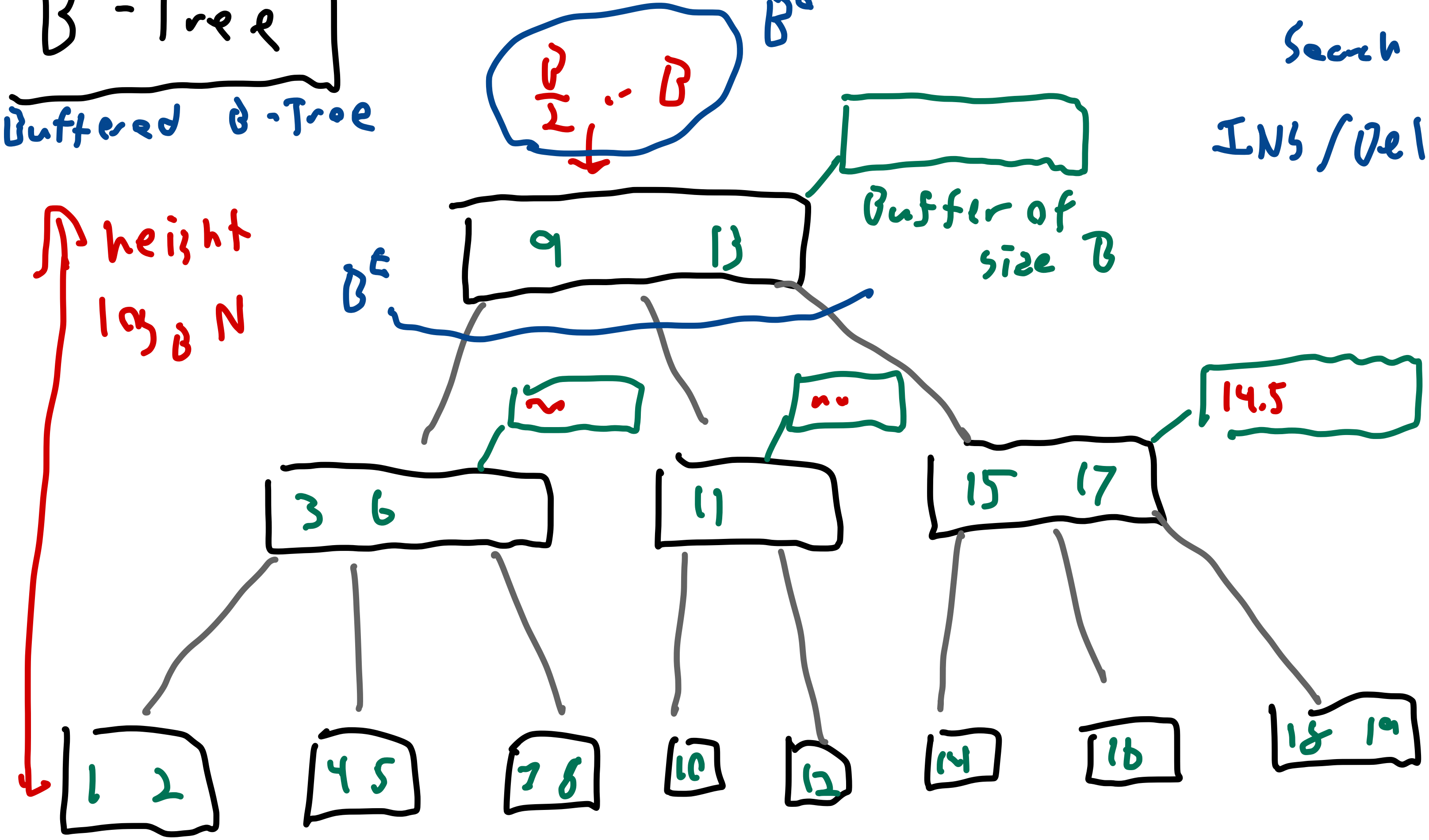




between $\frac{B}{2}$ .. $B$ pieces of data



# $B^{\epsilon}$-Tree

Buffered B-Tree




Search $O(\log_B N)$

INS / Del $O(\log_B N)$

B pieces of data

$B^{\epsilon}$ children "cost"

$\frac{B^{\epsilon}}{B}$ cost per data $= \frac{1}{B^{1-\epsilon}}$

- In Buffer, No ordering

Old Height $\log_B N$

New Height $\log_{B^{\epsilon}} N \approx \frac{1}{\epsilon} \log_B N$

Time for Insertion is $O\left(\frac{1}{B^{1-\epsilon}} \log_B N\right)$ old $O(\log_B N)$

$$\frac{1}{\sqrt{B}} \log_B N \quad \text{vs} \quad \log_B N$$

$< 50$

$$\frac{1}{\sqrt{B}} \frac{\log N}{\log B} \qquad \frac{\log N}{\log B}$$

100

12

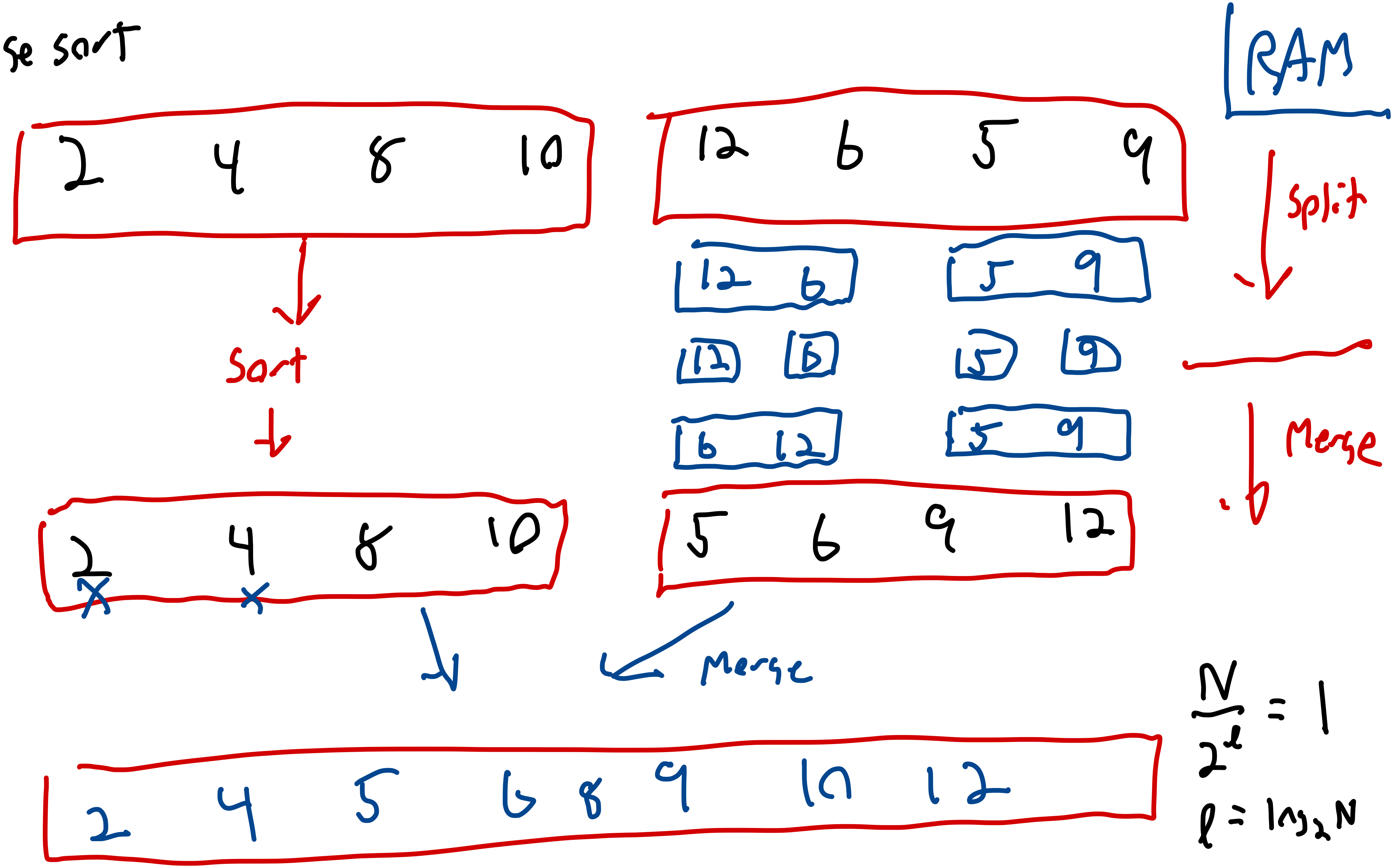
12

$B = 10000$

$\sqrt{B} = 100$

$$\frac{1}{24} \quad \text{vs} \quad 4$$

# Merge sort

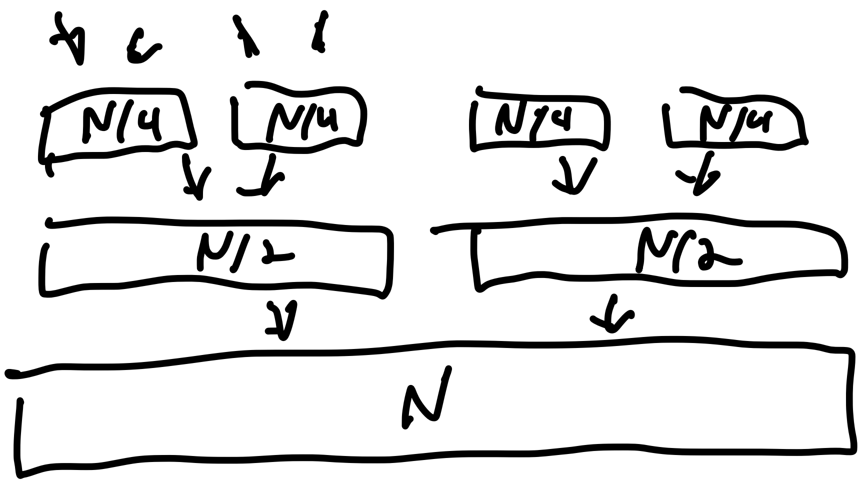



$$\frac{N}{2^{\ell}} = 1$$

$$\ell = \log_2 N$$

```
def MS(X)
    if len(X) > 1
        L = X[:len(X)//2]
        R = X[len(X)//2:]
        MS(L)
        MS(R)
        return Merge(L, R)
Else
    return X
```

1




$$\left.\begin{array}{l} N \\ N \\ \frac{N}{2}+\frac{N}{2}=N \\ N \end{array}\right\} \log_2 N$$

$$O(N \log_2 N)$$

12 1 2 5 10 15 30 70 11

Pivot

< pivot    > pivot

pivot

$\underbrace{1 \quad 2 \quad 5 \quad 10}_{\text{Recursively Sort}} \qquad 11 \qquad \underbrace{12 \quad 15 \quad 30 \quad 70}_{\text{Recursively Sort}}$

1 2 5 10 11 12 15 30 70

# Mergesort

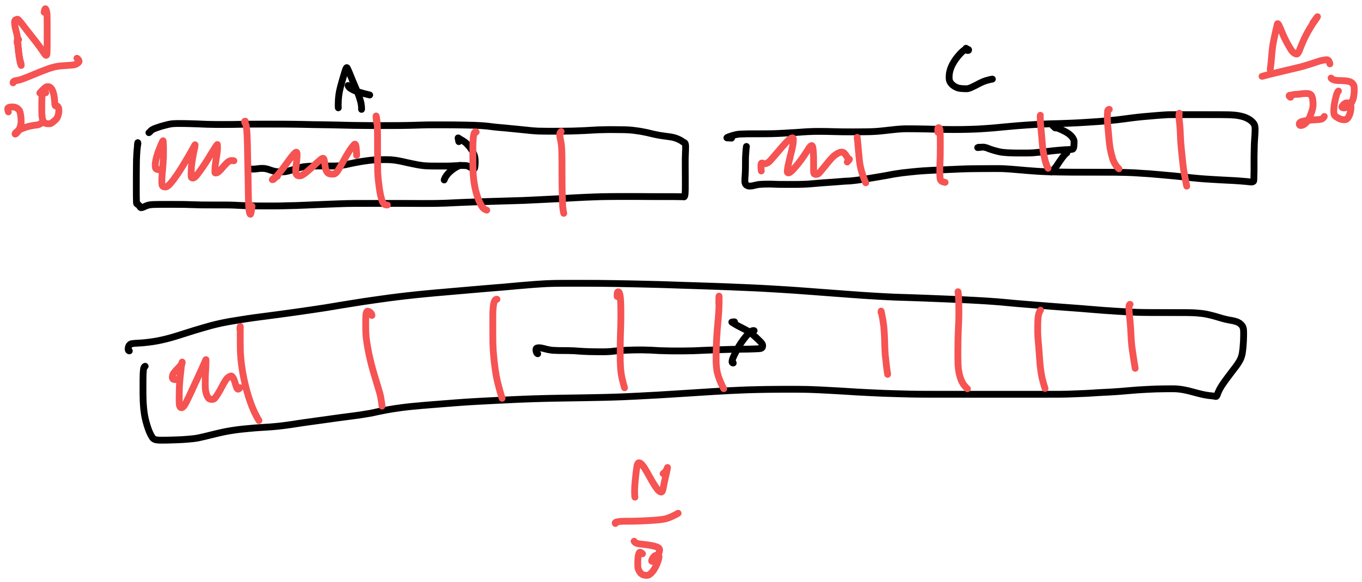
$$T(N) = 2T\left(\frac{N}{2}\right) + N$$

Mergesort

# What about DAM

## Sorting

| | RAM | DAM |
| --- | --- | --- |
| Mergesort | $O(N \log N)$ | $O(\frac{N}{B} \log N)$ |



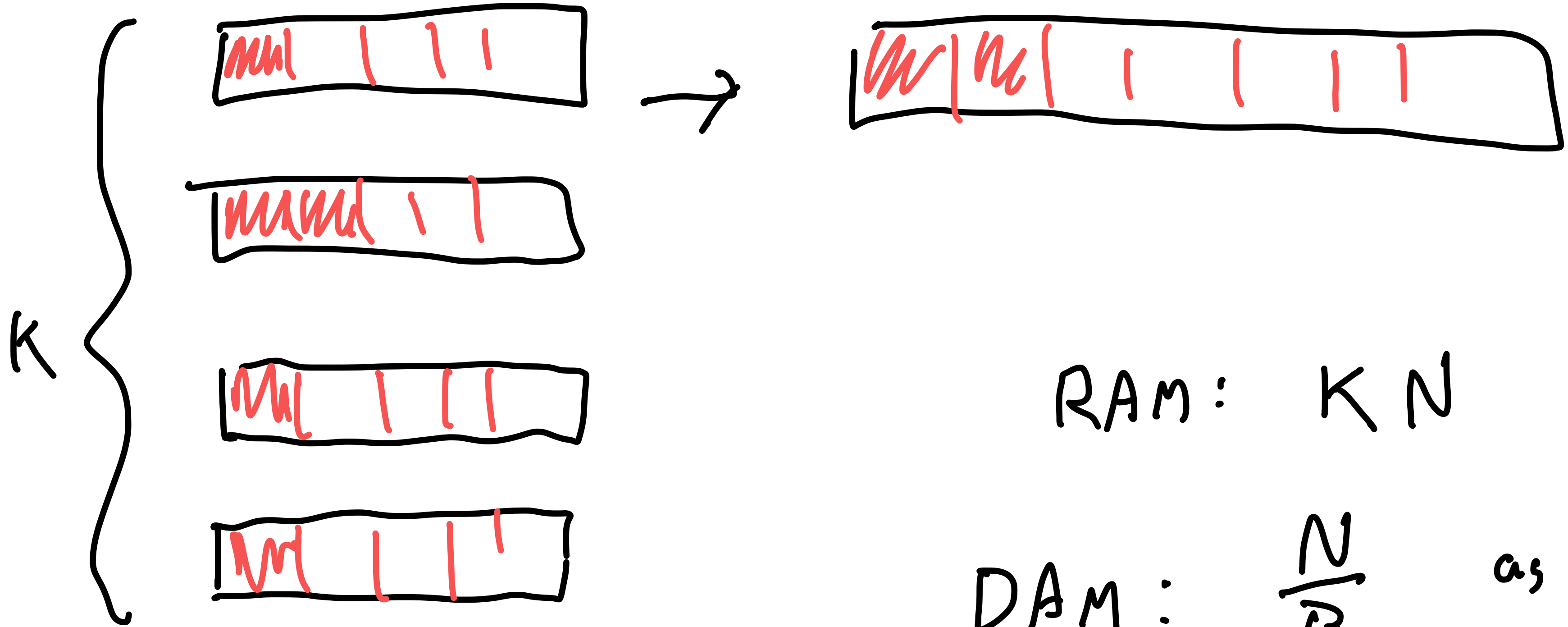

**Merge**

RAM $O(N)$

DAM $O(\frac{N}{B})$

K-way Merge

$\frac{N}{B}$



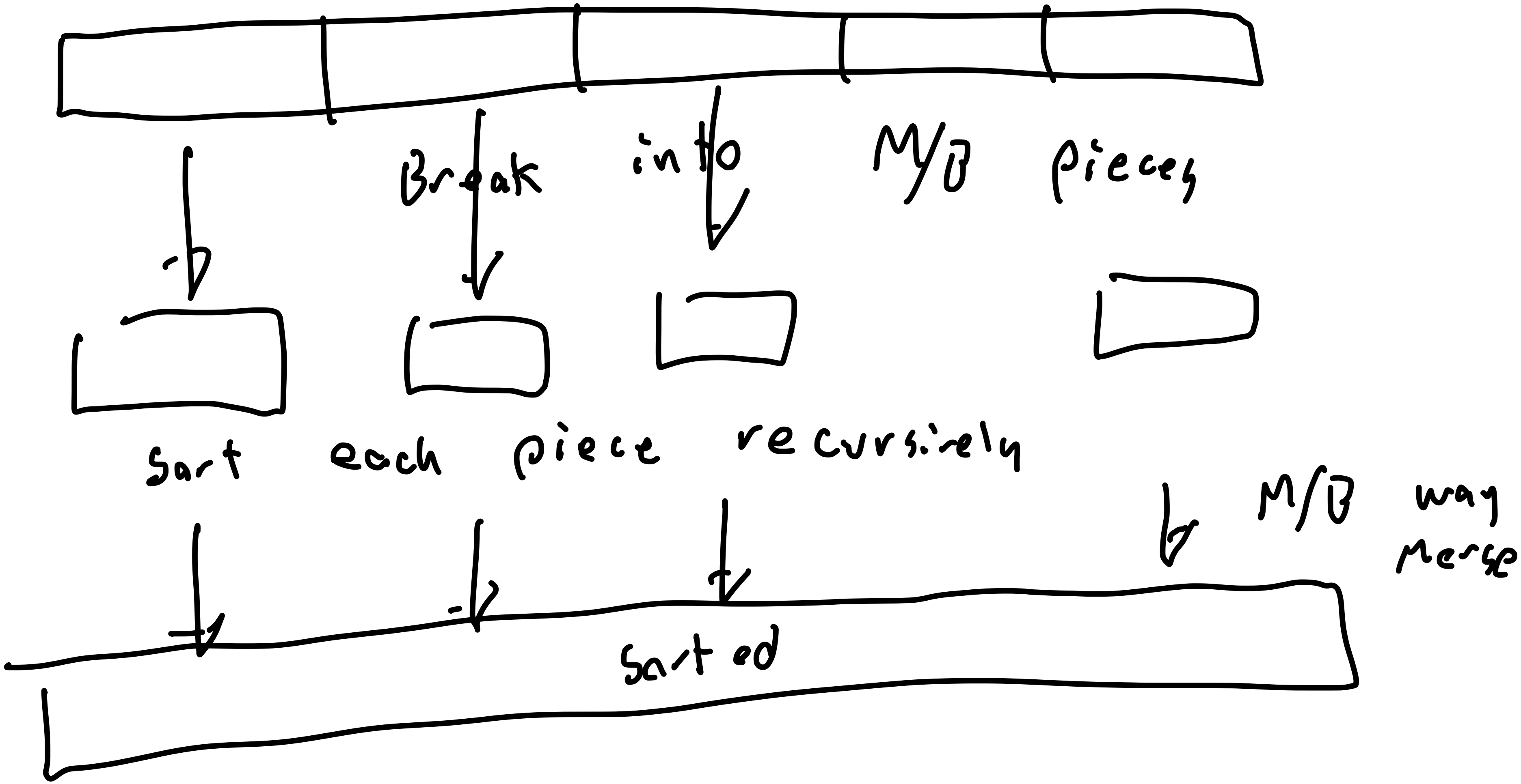

sorted lists total size is N

RAM: $KN$

DAM: $\frac{N}{B}$ as long as one block from each list fits into memory

$K \cdot B \le M$

$K \le \frac{M}{B}$




2 way Mergesort $O\left(\frac{N}{B}\log_2 N\right)$

$\frac{M}{B}$ way Mergesort $O\left(\frac{N}{B}\log_{\frac{M}{B}} N\right)$

faster by a factor of $\log \frac{M}{B}$

Billions

Thousands