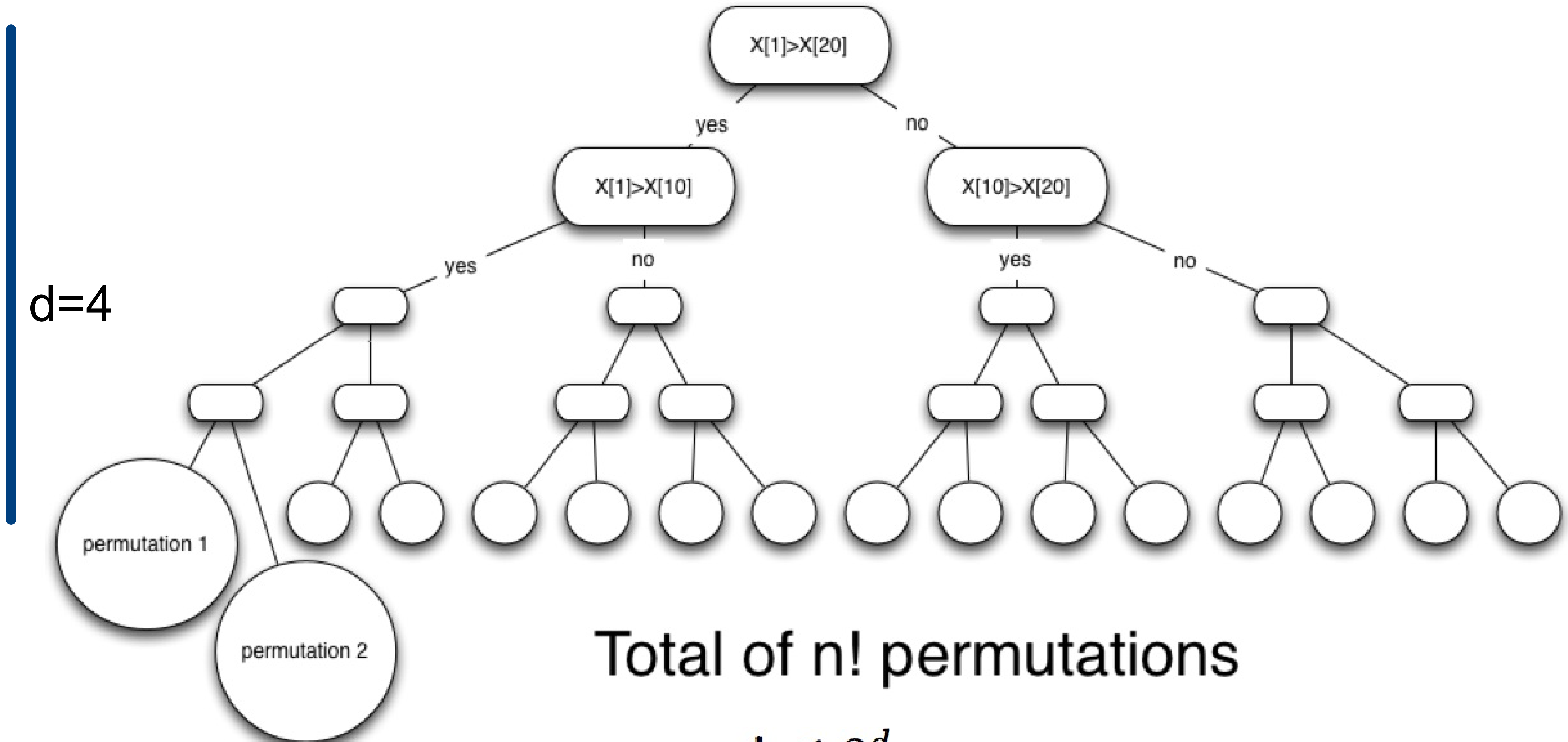
sorting requires n log(n) time in the worst case



Total of n! permutations

$$n! \leq 2^d$$

$$d \ln 2 \geq \ln n! \geq (n-1) \ln n$$

Sorting IID draws from the uniform distribution U(min,max)

Suppose we want to sort an array of numbers $S[1 \cdots n]$

- Divide $[\min, \max]$ into n equal-sized intervals. These are the *buckets* $B_1, B_2, \ldots, B_n$.
- Now scan array S from left to right, putting each element $S[i]$ in its appropriate bucket.
- Return $\text{sort}(B_1) \circ \text{sort}(B_2) \circ \cdots \circ \text{sort}(B_n)$, where "sort" is a standard sorting algorithm (say mergesort).

Let N_i be the number of array elements that fall into B_i. Assuming we use a standard sorting procedure for each bucket, we get a total running time of

$$T = N_1 \log N_1 + N_2 \log N_2 + \cdots + N_n \log N_n \;\leq\; N_1^2 + N_2^2 + \cdots + N_n^2.$$

What is $\mathbb{E}(N_i^2)$? The easiest way to compute this is to write N_i as a sum:

$$N_i = X_1 + X_2 + \cdots + X_n$$

where X_j is 1 if the array element $S[j]$ falls into bin i, and 0 otherwise. Notice that $X_j^2 = X_j$, and that X_j is independent of $X_{j'}$ whenever $j \neq j'$. Therefore,

$$\begin{aligned}
\mathbb{E}(X_j) &= \frac{1}{n} \\
\mathbb{E}(X_j^2) &= \frac{1}{n} \\
\mathbb{E}(X_j X_j') &= \mathbb{E}(X_j)\mathbb{E}(X_{j'}) \;=\; \frac{1}{n^2} \quad \text{if } j \neq j'
\end{aligned}$$

By linearity of expectation, we then have

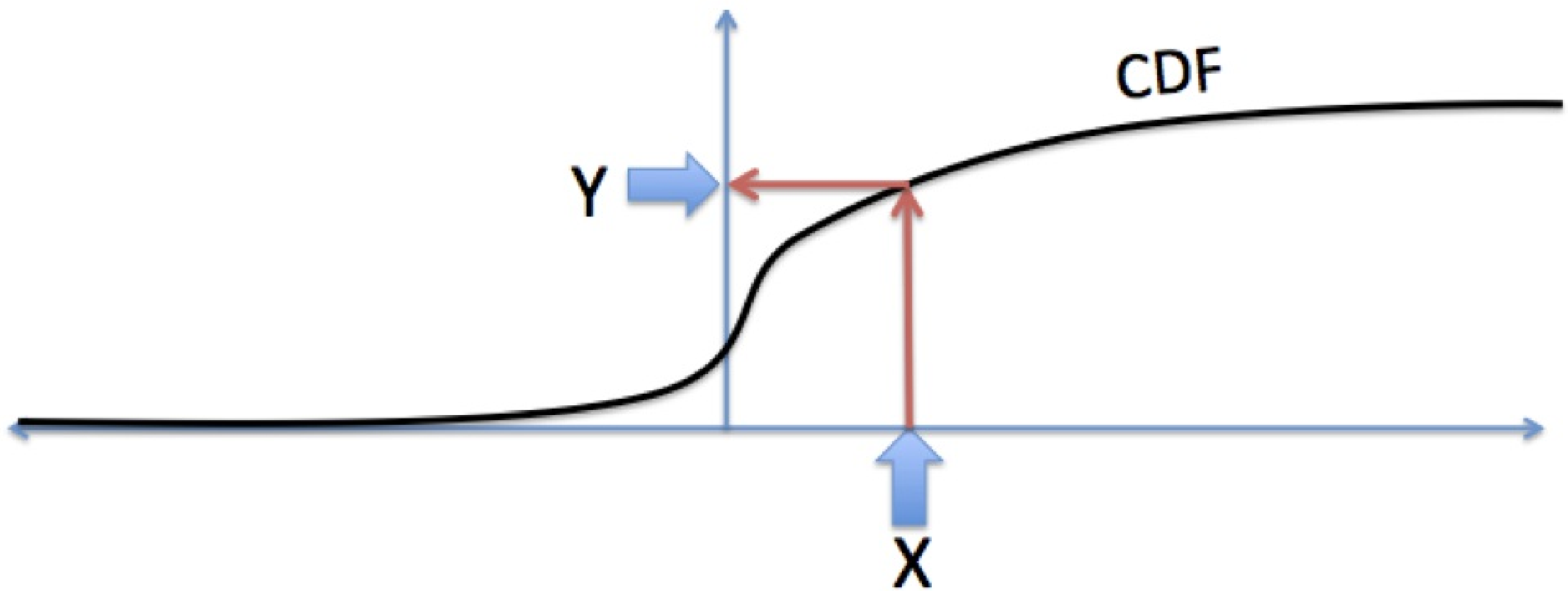
$$\begin{aligned}
\mathbb{E}(N_i^2) &= \mathbb{E}\left((X_1 + \cdots + X_n)^2\right) \\
&= \mathbb{E}\left(\sum_j X_j^2 + \sum_{j \neq j'} X_j X_{j'}\right) \\
&= \sum_j \mathbb{E}(X_j^2) + \sum_{j \neq j'} \mathbb{E}(X_j X_{j'}) \\
&= n \cdot \frac{1}{n} + n(n-1)\frac{1}{n^2} \ \leq \ 2.
\end{aligned}$$

So the expected running time of the sorting algorithm, once again invoking linearity, is

$$\mathbb{E}(T) \ \leq \ \mathbb{E}(N_1^2) + \mathbb{E}(N_2^2) + \cdots + \mathbb{E}(N_n^2) \ \leq \ 2n.$$

It is linear!

Sorting Non-Uniform distributions



Instead of sorting X, sort CDF(X)

Suppose X is distributed according to the CDF function $F(\cdot)$

By definition: $F(a) = \Pr(X \le a) = \Pr\left(F(X) \le F(a)\right)$

Therefore $\Pr\left(A \le F(X) \le B\right) = B - A \Rightarrow$ This is the uniform distribution U(0,1)

If we know the CDF, we can sort in linear time
Useful if we can compute CDF(X) in constant time.

Computing the CDF from a sample = Sorting

If all we have is a sorted sample then computing CDF(X) takes logarithmic time.

Everything works nicely if we have a functional representation of the distribution. Uniform, exponential normal, mixtures ...

Another use of the CDF: generating pseudo-random numbers with a known CDF

Generate number Y from uniform distribution

Use inverse of CDF to produce X