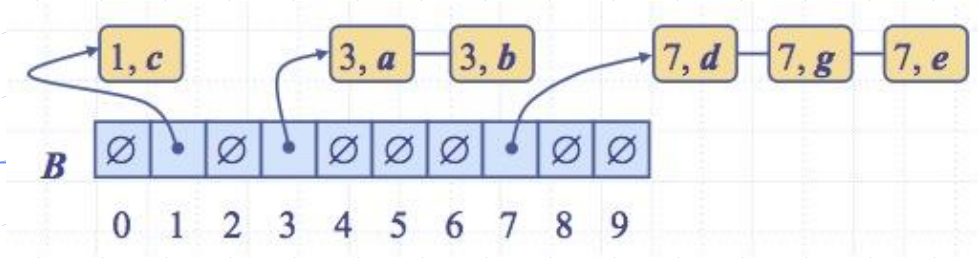
Lesson 7

Lower Bound on Comparison-Based Algorithms: *Discovering the Range of Natural Law*



Wholeness of the Lesson

Using the technique of decision trees, one establishes the following lower bound on comparison-based sorting algorithms: Every comparison-based sorting algorithm has at least one worst case for which running time is $\Omega(n \log n)$. Bucket Sort and its relatives, under suitable conditions, run in linear time in the worst case, but are not comparison-based algorithms. Each level of existence has its own laws of nature. The laws of nature that operate at one level of existence may not apply to other levels of existence.

Road Map for Lower Bound on Comparison Based Sorting

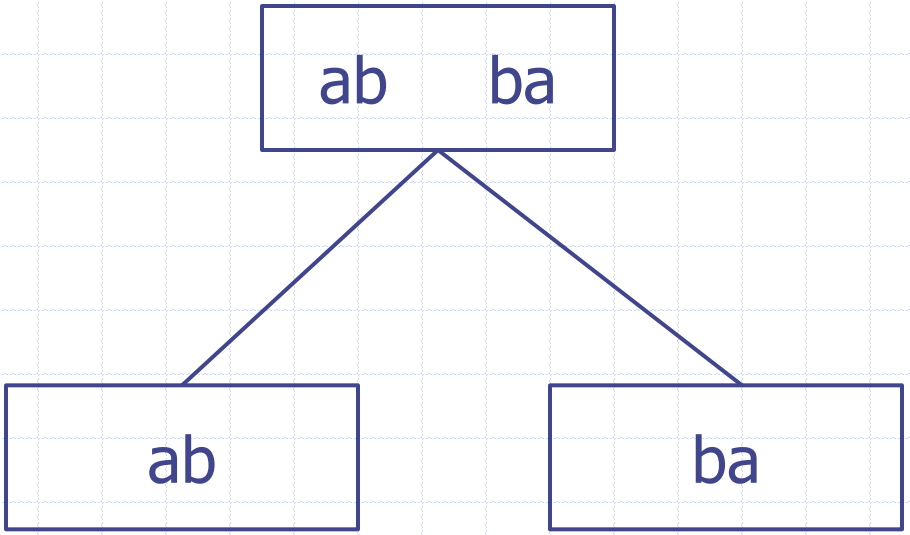
1. Given n, there are n! permutations.
2. Therefore any decision tree to sort n items will have n! leaves.
3. Since the decision tree is a binary tree, it must at least have log (n!) height.
4. Since the height of the decision tree is the same as the number of comparisons, any comparison based sorting algorithm must perform log (n!) comparisons.
5. log (n!) is theta(n log n).
6. Thus lower bound for the comparison based sorting algorithms is big-Omega(n log n)

Sample Decision Tree

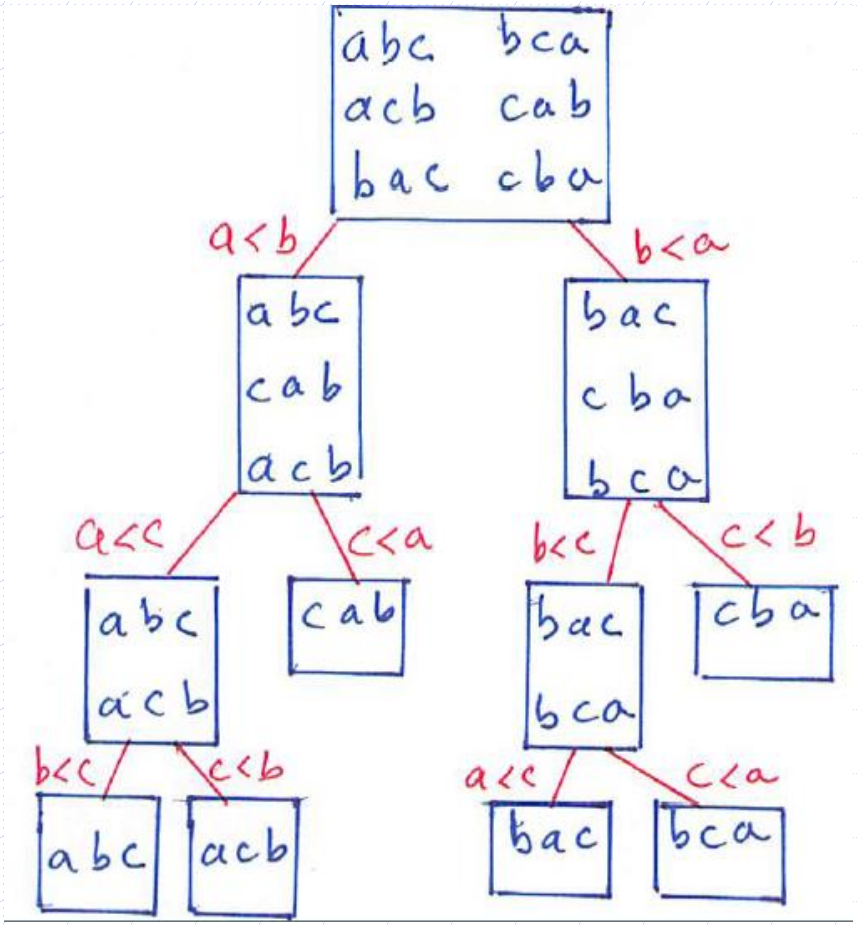
- Every comparison-based sorting algorithm, applied to a given array, can be represented by a decision tree
- To illustrate the technique, we use a simple but typical example: sorting 2, 3 and 4 element array of distinct values *a,b,c,d.*

Decision Tree for Sorting 2 Elements

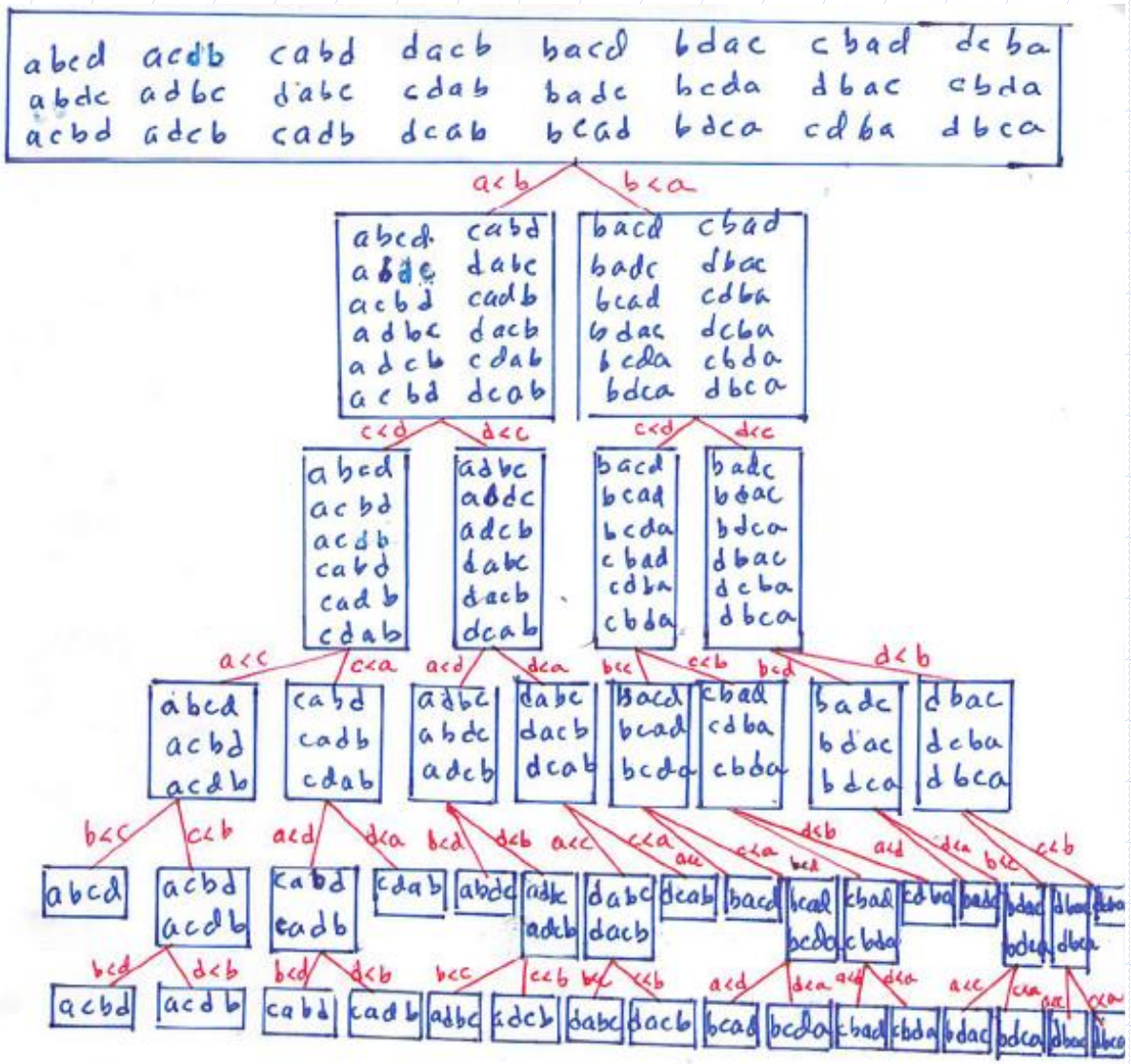
```
Read a, b
if (a < b)
    print(a, b)
else
    print(b, a)
```



Decision Tree for Sorting 3 Elements



Decision Tree for Sorting 4 Elements



Anatomy of the Decision Tree

- Each node of the tree represents all possible sorting outcomes that have not been eliminated by the comparisons done so far.
- The labels on the links of the tree represent comparison steps as the algorithm runs. Different algorithms will perform some of the comparison steps in a different sequence.
- A leaf in the decision tree represents a possible sorting outcome. The different paths to the leaves represent all possible ways three (as in slide 4 or four as in slide 5) distinct items could be put in sorted order; therefore, the leaves represent all possible arrangements of three (or four) distinct elements.

Strategy for Computing # Comparisons

- *Observation:* The number of comparisons performed in order to arrive at an arrangement found in a leaf node equals the depth of that leaf node in the decision tree.

- Therefore, to count # comparisons performed in the worst case, we determine the depth of the deepest node in the decision tree

Establishing the Lower Bound

Summary: Suppose we have a decision tree T for a sorting algorithm running on input of size n. Then T has n! leaves. So the height of T is at least $\lceil \log n! \rceil$ so some leaf in T has depth at least $\lceil \log n! \rceil$. Therefore, in the worst case, at least $\lceil \log n! \rceil$ comparisons are performed, so running time is $\Omega(\log n!)$. We determine the complexity class of log n!

Main Point

A decision-tree argument shows that comparison-based sorting algorithms can perform no better than Θ(nlog n). However, BucketSort is an example of a sorting algorithm that runs in O(n). This is possible only because BucketSort does not rely primarily on comparisons in order to perform sorting. This phenomenon illustrates two points from SCI. First, to solve a problem, often the best approach is to bring a new element to the situation (in this case, bucket arrays); this is the Principle of the Second Element. The second point is that different laws of nature are applicable at different levels of creation. Deeper levels are governed by more comprehensive and unified laws of nature.

Connecting the Parts of Knowledge
With The Wholeness of Knowledge

Transcending The Lower Bound On Comparison-Based Algorithms

1. Comparison-based sorting algorithms can achieve a worst-case running time of $\Theta(n \log n)$, but can do no better.

2. Under certain conditions on the input, Bucket Sort and Radix Sort can sort in $O(n)$ steps, even in the worst case. The $n \log n$ bound does not apply because these algorithms are not comparison-based.

***3. Transcendental Consciousness* is the field of all possibilities and of pure orderliness. Contact with this field brings to light new possibilities and leads to spontaneous orderliness in all aspects of life.**

4. *Impulses Within The Transcendental Field*. The organizing power of pure knowledge is the lively expression of the Transcendent, giving rise to all expressions of intelligence.

5. *Wholeness Moving Within Itself*. In Unity Consciousness, the organizing dynamics at the source of creation are appreciated as an expression of one's own Self.