# CS591A1 Spring 2019 - Research Project

**Title:** *Performance analysis of B+Trees and Tries with variable workload sortedness*

**Background**: Data, in many cases, comes with *implicit ordering*, i.e., the incoming data set comes implicitly partitioned for some attributes which are correlated with time. This implicit ordering of data may cause algorithms to perform sub-optimally in several cases. For example, if the data comes in sorted order they trigger the worst-case runtime ($O(n^2)$) for a quick sort routine. Similarly, the insert cost for new entries may vary from O(log n) (for completely scrambled datasets) to O(1) (for fully sorted datasets), depending on the *orderness* (pre-sortedness) of the datasets and the access methods used to insert the data. A study of the effects of implicit ordering within workloads on performance of the access methods, is therefore, the first step towards the design of sortedness-aware access methods.

**Objective**: This project is designed to be implementation-heavy, the objective of which is to study the performance of the classical and advanced access methods against workloads with implicit ordering and also against completely scrambled workloads. Following is the workflow, which must be adhered to in course of this project.

(a) Study the variation in the read and write costs of B+ Trees and Tries [2] for workloads with variable degree of pre-sortedness
(b) Build a benchmark and measure their performance (read/write cost, main memory footprint) against workloads with variable degree of pre-sortedness

**Benchmark workloads**: Targeted micro-benchmarks and TCP-H, YCSB .

[1] Heikki Mannila. **Measures of Presortedness and Optimal Sorting Algorithms**. IEEE Trans. Computers 34(4): 318-325 (1985)
[2] Viktor Leis, Alfons Kemper, Thomas Neumann. **The adaptive radix tree: ARTful indexing for main-memory databases**. ICDE 2013: 38-49