A conventional algorithm uses a single processing element. A parallel algorithm assumes that there are multiple processors. These processors may communicate with each other using a shared memory or an interconnection network. An algorithm designed for a large number (for example, a polynomial in the problem size) of processors can be simulated on a machine with a small number of processor for a trade off on time, and therefore is of practical value, while at the same time allowing us to test the limits of parallelism. Many algorithmic design techniques in the parallel setting will be explored. Parallel complexity theory will also be briefly studied.


**COURSE OUTLINE:**

**A conventional algorithm uses a single processing element. A parallel algorithm assumes that there are multiple processors. These processors may communicate with each other using a shared memory or an interconnection network. An algorithm designed for a large number (for example, a polynomial in the problem size) of processors can be simulated on a machine with a small number of processor for a trade off on time, and therefore is of practical value, while at the same time allowing us to test the limits of parallelism. Many algorithmic design techniques in the parallel setting will be explored. Parallel complexity theory will also be briefly studied.**

**ABOUT INSTRUCTOR:**


**COURSE PLAN:**

**Week 01:** Theoretical models of parallel computation: PRAM, interconnection networks

**Week 02:** Performance of parallel algorithms, Basic techniques

**Week 03:** Basic techniques (cont’d)

**Week 04:** Comparator Networks. Odd Even Merge Sort, Bitonic Sort, Merge Sort.

**Week 05:** Optimal List ranking, applications

**Week 06:** Algorithms for searching, merging and sorting. Cole’s Merge Sort

**Week 07:** Cole’s Merge Sort (cont’d), Graph algorithms

**Week 08:** Graph Algorithms (cont’d), Linear Array, Meshes

**Week 09:** Sorting in meshes, Hypercube algorithms, Butterfly network, CCC, Benes network

**Week 10:** Butterfly network, ccc, benes network

**Week 11:** Limits to parallelizability. Lower bounds

**Week 12:** Limits to parallelizability. NC-reductions, P-completeness.