Project List

1. Additya
    1.1. **MPI/Erlang:** Implement parallel Monte Carlo simulation. Explore different methods for generating random numbers in parallel.
    1.2. **Erlang:** Study how BFS algorithms can be implemented in Erlang. Implement them. Compare their performances.
    1.3. **MPI/Erlang:** Implement a program to solve a given system of linear equations in the form Ax=b using the Gaussian Elimination algorithms.
    1.4. **MPI/Erlang:** Implement a program to solve a given system of linear equations in the form Ax=b using the Conjugate Gradient Method.
    1.5. **MPI/Erlang:** Implement a distributed/parallel program to solve the stable marriage problem.
2. Aman
    2.1. Build a scalable web cache using **Consistent Hashing**.
    2.2. Experiment with various **Path Pushing** deadlock detection algorithms under various message delivery guarantees and study the number of messages required to detect the deadlock(s).
    2.3. Experiment with various **Edge Chasing** deadlock detection algorithms under various message delivery guarantees and study the number of messages required to detect the deadlock(s).
    2.4. Experiment with various **Diffusion Computation** deadlock detection algorithms under various message delivery guarantees and study the number of messages required to detect the deadlock(s).
    2.5. Experiment with various **Global State** deadlock detection algorithms under various message delivery guarantees and study the number of messages required to detect the deadlock(s).
3. Avinash
    3.1. Simulate message delivery guarantees such as **Causal and Arbitrary**, and their impact on some **Termination Detection** distributed algorithm.
    3.2. Simulate message delivery guarantees such as **FIFO and Arbitrary**, and their impact on some **Termination Detection** distributed algorithm.
    3.3. Simulate message delivery guarantees such as **Causal and Arbitrary**, and their impact on some **Mutual Exclusion** distributed algorithm.
    3.4. Simulate message delivery guarantees such as **FIFO and Arbitrary**, and their impact on some **Mutual Exclusion** distributed algorithm.
4. Devansh
    4.1. **Distributed Storage**. Performance bottlenecks and challenges faced.
    4.2. **Distributed Verification**. Performance bottlenecks and challenges faced. Propose your own project by preparing a one-page writeup including the problem description and the deliverables. Once your project proposal is approved by the instructor, you can go ahead.

    4.3. Experiment with various **Leader Election** algorithms for various topologies and study which algorithms would perform best (in terms of number of messages) for different topologies.
    4.4. **Erlang:** Study the weighted byzantine agreement problem and then implement an algorithm for the weighted byzantine agreement problem. Study its performance and the algorithm for dynamic weight update. Also write about the challenges you faced.
    4.5. Implement a basic distributed MapReduce framework. Handle worker failure. (https://static.googleusercontent.com/media/research.google.com/en//archive/mapreduce-osdi04.pdf)
5. Karandeep
    5.1. **Implement GFS** with additional features such as replication of the master.
    5.2. **Verifying the MST**: The input is a weighted graph G and a tree T with the claim that T is an MST of G. Read the MST verification algorithms for distributed computing and implement such an algorithm using MPI.
    5.3. **Chord**: Implement the P2P network Chord and support Insert/Delete/Search operations.
    5.4. **Enumerative algorithms on Graphs** Counting number of triangles and 4-cycles using MPI
    5.5. **Enumerative algorithms on Graphs** Counting number of triangles and 4-cycles using Giraph
6. Pratik
    6.1. **Enumerative algorithms on Graphs** Counting number of triangles and 4-cycles using GoFFish
    6.2. **Erlang:** Study how Graph Connectivity algorithms can be implemented in Erlang. Implement them. Compare their performances.
    6.3. **Erlang:** Study how Graph Coloring algorithms can be implemented in Erlang. Implement them. Compare their performances.
    6.4. **Erlang:** Study how Matrix Multiplication algorithms can be implemented in Erlang. Implement them. Compare their performances.
    6.5. **Blockchain** and distributed systems. Performance bottlenecks and challenges faced.