This document indicates scope for 43 projects. These 43 projects are grouped into three main classes: MPI Collective operations, DHTs, File Systems, and Algorithms.

1. MPI Collective Operations

One of the important category of MPI supported operations is the notion of collectives. These include MPI_REDCUE, MPI_SCATTER, MPI_GATHER, MPI_ALLtoALL, and so on. See an MPI guide for the entire list of collectives and their semantics. Implementing these operations on a particular target network poses interesting implementation challenges. For instance, if the target network is a tree, one can use ideas such as moving information up and down the tree. If the target network is a mesh (aka grid), we may use row-wise or column-wise communication.

The goal of this group of projects is to pick two or more collective operations and two topologies from the following list of target network topologies. For the chosen operations and topologies, the project would design algorithms to support the collective operation in the most efficient way possible. Efficiency is to be measured as the time taken for the collective, the number of messages and rounds involved.

Operations:

a) Allgather
b) Allreduce
c) Alltoall
d) Bcast
e) Gather
f) Reduce
g) Reduce_scatter
h) Scan
i) Scatter,

Topologies:

a) Two-dimensional mesh of n rows and n columns
b) Three-dimensional mesh of n rows, n columns, and n height
c) A hypercube of k dimensions with n nodes, n being a power of 2. In this case, k is $\log_2 n$.

We will have projects 1 to 10 covering this space with different choices of operations and topologies. In the form for filling the project, in the box for any other comments, you may fill the choice of operations and topologies.

2. DHTs

One of the good projects in distributed systems is to implement various kinds of DHTs. In these projects, the goal is to implement a DHT as per the original design and test the system at scale. To this end, the system should be able to store millions of objects over thousands of machines. The search method and a possible application is what is studied for performance and scalability in addition to processing the join/leave of machines. In most cases, the entire system design is available as part of existing papers.

On top of the baseline system, the project should attempt to add at least one feature that is not part of the baseline system. For instance, one can think of replication, fault tolerance, consistency, heterogeneity, and the like.

The project essentially is therefore a choice of an existing system and a choice of an enhancement. This allows for multiple projects to be supported from the following list.

Systems:

1. DynamoDB
2. Chord
3. Content Addressable Network (CAN)
4. Tapestry
5. Napster

Enhancement:

a. Replication
b. Fault tolerance
c. Consistency
d. Heterogeneity
e. Your own idea here.

Project numbers for this start from 11 and go as follows. Projects 11 to 15 are for DynamoDB plus one enhancement. Projects 16 to 20 are for Chord plus one enhancement, and so on. These projects go from numbers 11 to number 35. You can enter the project number accordingly.

3. File Systems

Another set of projects that are a good way to understand distributed systems in practice is file systems. These offer scope for setting up a basic system and propose additional enhancements. In that spirit, we studied two simple file systems in class: GFS and Haystack. For each of these, we could think of various enhancements, some of which are listed below.

1. GFS:
    a. dynamic replication
    b. exactly once semantics for RecordAppend
    c. replicas by geography
    d. multiple master nodes
    e. Your own idea here
2. Haystack
    a. Dynamic replication
    b. Link to similar photos (use some similarity measure to link from one photo to another very similar photo, either in the same collection or across collections)
    c. Your own idea here.

Like in the case of DHTs, the projects on GFS get numbers 36 to 40 and the projects on Haystack get numbers 41 to 43.

Graph Algorithms

Another popular project topics in distributed systems is to study distributed algorithms on a variety of distributed computing models. Typically, graph problems are studied in this context. In the following, we propose four graph problems and six different models/platforms for implementing algorithms for these graph problems. Each project can pick a problem and two platforms to study algorithms for the chosen problem in the two platforms and do a comparative analysis with respect to time taken, number of communication rounds, message complexity, and so on.

Graph Problems:

1. Graph Coloring
2. Minimum Spanning Tree
3. Graph maximal matching
4. All-pairs-shortest-paths

Model/Platform

a. MPI
b. K-Machine
c. Massively Parallel Computing (MPC) linear regime
d. Massively Parallel Computing (MPC) sub-linear regime

As earlier, these give rise to another 24 projects based on the choice of the problem and the two platforms. We can go with number 44 for the choice of graph coloring on models MPI and k-machine, number 45 for the graph coloring on models MPI and MPC-linear

regime, and finally project number 67 for the APSP problem on MPC linear regime and MPC sub-linear regime.

Project 68: Create your own project – to send me a description for scoping. When filling the form, you may enter 68 as the first choice and mention another choice just so that we have a backup.

Reference Material:

For all the projects, there is enough reference material that is easily available on the web. If you face any questions, please contact me by email or in person. I can suggest some good references.

Programming and Testing:

Each project will involve some programming component and extensive and scalable testing. In particular, testing should be done on large datasets/ graphs as the case may be. For graphs, one useful collection of datasets is the University of Florida Dataset.

Platform:

Except for specific projects that use MPI or Map-Reduce, others can be simulated on a single machine. For projects on MPI etc., we will provide the required platform once we know the groups who are executing such projects.

Timelines and Deliverables:

Each project team will have the following deliverables spread across October to the final submission.

1. A 2-page report on the project scope, problem being solved, basic ideas, and the like to be submitted by October 14, 2025 on moodle. While you do not have to wait till then to start working on your project, this extended time is provided keeping in mind that HW3 is just out. This submission counts as Homework 4.
2. A mid-evaluation of the project based on another 2-page submission will be due early November 2025. Date will be announced soon.
3. Final project evaluation will be an in-person assessment with each group getting a 15 minute slot. The assessment will require a presentation, a report to be uploaded prior to the assessment, all the software developed, and plots produced, etc. to be uploaded prior to the assessment. The date/slots for the

assessment will be announced closer to the date of the final examinations. These slots will be just after the week of the exams.