SCRIPT:

Here we will be presenting the research paper - “An Umbrella Converse for Data Exchange: Applied to Caching, Computing, and Shuffling “

Saumya:

Model- The model of the paper is to study the problem of data exchange in a networked environment The paper aims to derive lower bounds and converses for various settings and scenarios with a focus on the application of coded caching, computing, and shuffling using mathematical and information-theoretic techniques.

Himani:

Problem – The problem that the author is trying to solve in this paper is the efficient exchange of data between multiple nodes in a network. The author aims to design communication schemes that minimize the amount of data transmitted while ensuring that each node has access to the desired data. The author's specific goal is to develop a converse to the data exchange problem, which provides an expression that depends only on the number of bits to be moved between different subsets of nodes.

Saumya:

Nodes and data exchange problem:

- In the context of data exchange, nodes refer to individual entities within a network that participate in the exchange of data. Nodes communicate with each other through noiseless bus links (where transmission of any node is received by all others).
- As per Data exchange problem, Each node possesses a collection of data symbols (represented in bits) in its local storage(c_i) and demands another set of symbols present in other nodes(d_i)

Definition 1. *A data exchange problem on a set of K nodes involving a collection B of information bits is given by the following:*

- *a collection $\{C_i : i \in [K]\}$, where $C_i \subset B$ denotes the subset of data present in node i,*
- *a collection $\{D_i : i \in [K]\}$ where $D_i \subset \cup_{j \neq i} C_j \setminus C_i$ denotes the set of bits demanded by node i.*

.

- The goal is to devise a mechanism or algorithm to efficiently exchange the required bits among the nodes to meet their respective data demands.

Himani:

- A solution to this given data exchange problem involves communication between the nodes. Each node i encodes the symbols in Ci into a codeword of length li and sends it to all other nodes. The respective demanded symbol at any node is then to be decoded using the received transmissions from all the other nodes and the node's own content.
- The objective is to enable each node to acquire the bits it demands from other nodes while minimizing the overall communication load or maximizing the efficiency of data transmission.
- As per the described scheme, communication load is defined as the total number of bits communicated --- formula----and the optimal communication load is-------

Saumya:

- Here we will obtain a lower bound (a minimum achievable amount of communication required to accomplish a particular task ((here the exchange is of information bits among the nodes in the network)) on the optimal communication load of the general data exchange problem as defined.
- Given a data exchange problem and for P, Q ⊂ [K] such that P 6= ∅, let a Q P denote the number of bits which are stored in every node in the subset of nodes Q and stored in no other node, and demanded by every node in the subset P and demanded by no other node, i.e.,
- Note that, by definition, a Q P = 0 under the following conditions. • If P ∩ Q!= ∅, as the bits demanded by any node are absent in the same node. • If Q = ∅

Himani:

- Theorem 1 gives a lower bound on the optimal communication load of a given data exchange problem.
- IDEA OF THE PROOF: -
  If we consider only two nodes in the system, say [K] = {1, 2}, then each of the 2 nodes has to transmit whatever bits it has which are demanded by the other node, i.e., L ∗ ≥ a {1} _ {2} + a {2} _ {1}
- Meaning of idea: - The term a {1}{2} represents the number of bits that node 1 demands from node 2 and are exclusively stored in node 2. Similarly, a {2}{1} represents the number of bits that node 2 demands from node 1 and are exclusively stored in node 1. To achieve the optimal communication load, both nodes need to transmit the bits that are demanded by the other node, as there is no other way for the demanded bits to reach their respective nodes. Thus, the lower bound for the optimal communication load is given by the sum of these two quantities, a {1}{2} + a {2}{1}.

Saumya:

- The three steps which describe the generic structure behind our converse proofs are:
  Applying theorem 1, we obtain a lower bound expression on the communication load, assuming an arbitrary choice of demands across the nodes and some arbitrary but fixed storage across the nodes.
  This step helps to remove the dependency of the lower bound on the specific choice of demands. Here, the lower bound expression obtained is averaged over some carefully chosen configurations of demanded bits at the nodes.
  The objective of these refinements is to remove the dependency of the converse on the specific initial storage configuration at the nodes. This is done by imposing the constraints on the size of the initial storage at the nodes and using convexity of terms inside the averaged bound to obtain the final expression of the bound. This allows for a more comprehensive analysis of the data exchange problem.

Himani:

The above is used to recover lower bound for the problem of:
(a) Coded caching
(b) Decentralized coded data shuffling
(c) Coded distributing computing

Here we will mainly focus on Coded distributing computing:

N FILES ON WHICH THE DISTRIBUTED COMNPUTING TASK HAS TO BE PERFORMED BY K NODES.

There are N files and a subset Mi of them are assigned to node i in map phase. The output of map phase at each node is v1:Q,Mi . Each node computes Xi = ϕi (v1:Q,Mi) which it broadcasts to other nodes. The nodes compute the reduce outputs based on their own map outputs and the broadcasts which they receive.

- MAP -> A subset M_i of N files is assigned to ith node and the ith node computes the map functions on this subset in the map phase.We assume that the total number of map functions computed at the K nodes is rN, where r is referred to as the computation load.
- The input data is divided into multiple chunks, and each chunk is processed by a different node in parallel. Each node is responsible for executing the map function on its assigned portion of the input data.
- As a result, each node only has access to the intermediate outputs (key-value pairs) that are generated from the input data it processed. These intermediate outputs are stored locally on the node's disk.

Saumya:

1) Reduce -> a total of W reduce functions is to be computed across the K nodes corresponding to the N files. Each node is assigned the same number of functions. Obtaining the output of the

reduce functions at all the nodes will complete the distributed computing task. Now, when it comes to the reduce phase, each node is assigned a specific reduce function to execute. The reduce function takes the intermediate outputs from the map phase and performs further processing to produce the final output.

- In order to compute the assigned functions, each node requires access to all the intermediate outputs generated in map phase. However, since the map phase was executed in parallel across multiple nodes, each node only has the intermediate outputs that were generated from the input data it processed during the map phase. This means that each node is missing the intermediate outputs of those files that were not assigned to it during the map phase. Those missing intermediate outputs are present on the disks of other nodes in the cluster.

Himani:

- To overcome this, the MapReduce framework incorporates a shuffle phase. During the shuffle phase, the framework transfers the required intermediate outputs from the nodes that generated them to the nodes responsible for the reduce tasks. This ensures that each node has access to all the necessary intermediate outputs to perform its assigned reduce function.

In Shuffle, Nodes that are assigned to perform distributed computing task – exchange data. The authors proposed coded communication during shuffle phase to reduce the communication load. A coded scheme, named "coded distributed computing" (CDC), is proposed to demonstrate that increasing the computation load of the Map functions by a factor of r can create novel coding opportunities that reduce the communication load by the same factor.i.e. communication load is inversely proportional to computation load.

Saumya:

Considering the scenario where each reduce function is computed by exactly one node.

L_{DC}^{*} = MINIMUM COMMUNICATION LOAD -> It is the minimum number of bits broadcasted by K nodes in the shuffle phase minimised over all possible map functions assignments, reduce function assignments, and shuffling schemes, with a computation load r

Considering the scenario where each reduce function is computed at s nodes.

Himani:

main result - We have presented an information-theoretic converse result for a generic data exchange.

problem, where the terminals contain some data in their local storage(C_{i}) and want other data.

available at the local storage of other nodes(D_{j}, where i\neq j). The main result of this paper is the derivation of a converse to the data exchange problem, which provides a lower bound on the optimal communication load for various scenarios related to coded caching, computing, and shuffling.