CHAPTER 4
LINEAR-CODE MULTICAST ON PARALLEL ARCHITECTURES (LCM–PA)

This chapter proposes Linear-Code Multicast on Parallel Architectures (LCM–PA). It is well-known that parallel architectures intend fast receiving of complete information at several nodes and reduces the execution time by dividing task between various nodes. This practice requires high communication (as data from diverse nodes travels simultaneously) and computation (as diverse nodes perform different operations on data simultaneously). Several architectures have been proposed to overcome the problem of high communicational and computational time complexity for transferring and receiving information. In this chapter, to reduce the complexity of such communication, Linear Network Coding (LNC) is implemented in parallel environment. For verification of the approach some parallel architecture are considered for implementing network coding approach and the results are examined on these networks in a generic environment. Further a standard approach for parallel networks is formulated, which shows that, by applying this approach effect of faulty nodes, information size and communication complexity exponentially decreases with code length.

4.1 Information Rate in Parallel Architectures

Parallel architectures are the means to enable fast and efficient communication. It reduces time to complete a task as several processing units perform parallel execution. These architectures distribute large problems on several processing units and solves in less time than any single processor system. To utilize algorithms for efficient and fast communication, several interconnection networks have developed. In topologies with multiple nodes taking part at a time, the issues like faulty nodes (receiving multiple data from multiple sources), information size and communication complexity are prominent. Data communication needs better approaches to 1) remove these issues and 2) utilize processing units. Linear Network Coding (LNC) evolved as an efficient approach for data communication [1, 69, 120-123].
The information rate from source node to sink can potentially become higher when coding scheme is wider [1]. Further, in [1] by linear coding alone, the rate at which a message reaches each node, can achieve the individual max-flow bound. And provide a realization of the transmission scheme for both acyclic and cyclic networks [1].

We have considered a general approach for parallel multicast in multisource networks which is possible with correlated sources (figure 4.1). In each incoming transmission from source, the destination has knowledge of overall linear combination of data (see figure 4.1). This information updates at each coding node by applying the same linear mapping to the coefficient vectors as applied to the information signals. As an example, in a directed parallel network (Ą) (as in figure 4.1) set of two bits (d₁, d₂) is multicast from the source node P₁ (unique node, without any incoming at that instant of time) to other nodes of network. Figure 4.1 shows the network coding at each node, when information is multicast in a network. So, let us explain network coding using a network Ą as given in figure 4.1(a). At P₁, the data set (d₁, d₂) is multicast to destination nodes P₃ and P₇. The transmission progresses as: (d₁, d₂) traverse via node P₂ and P₄ nodes of network Ą; P₂ receives d₁ and P₄ receives d₂. Then both P₂ and P₄ transfer the information to P₅. Now as P₅ receives two diverse data from different sources. To avoid data loss P₅ perform network coding (d₁ ⊕ d₂). Node P₅ and P₆ send this encoded data to P₃ and P₇ which is decodes to receive the data (d₁, d₂).

This approach shows that multicast of different data is possible when network coding is used in the network. It is right to say that flow of information and flow of physical commodities are two diverse ideas [1]. So, the coding information does not increase the information
content. Capacity of network to transmit data from the source to destination can become higher if the coding scheme becomes wider, but it limits to max-flow for a wide coding scheme [1]. We have carried out this approach with parallel networks, and it results in a novel efficient manner of data communication. For parallel transmission of information, the linearity of code makes encoding (at source end) and decoding (at receiving end) easy in practice.

4.2 Review stage for network coding

In figure 4.1, we have considered a single source acyclic network to multicast two bits $d_1$ and $d_2$ to non-source nodes. However, in parallel networks there are multiple sources to multicast data to other several and diverse non-source nodes. In parallel networks, nodes may act as source and non-source nodes at different step of an algorithm. Parallel networks are acyclic at each step of an algorithm thus needs network coding. We have considered the approaches and examples of Li et al. [1] to fetch-out basic needs of network coding and justify our implementation on parallel networks. To follow the needs for network coding, let us consider network as in figure 4.1(a) in which $P_1$ is a source node and $P_2$, $P_4$, $P_5$ and $P_6$ are routing nodes between $P_1$ and $P_3$, $P_7$. Where, $P_6, P_3$ stands for a channel between node $P_6$ and $P_3$.

Parallel networks consist of busy channels in the flow. These are 1) channels that do not form directed cycles, (for example figure 4.1(a) and consider any sub-network between $P_1$ and $P_3$, $P_7$, let the sub-network is $P_1$, $P_2$, $P_5$, $P_6$, $P_3$ which is acyclic) 2) For nodes except $P_1$ and $P_5$, the number of incoming busy channels is equal to outgoing busy channels 3) The number of outgoing channels to $P_1$ is equals to number of incoming channels to $P_5$. This shows that if information rate increases, the rate of network coding also increases [1]. For parallel architectures having several coded nodes, communication using network coding may become impractical. This setback of parallel networks solves using Max-Flow law of information flow [124] (minimize the number of nodes to be coded, despite the acyclic constraint). According to Max-Flow Min-Cut Theorem, for every non-source node $P_2$ (figure 4.2 and 4.3 for explanation of Max-Flow Min-Cut on network $N$ and $M$) the minimum value of all cuts between $P_1$ and $P_2$ is equal to maxflow($P_2$) [124, Ch. 4, Theorem 2.3].
Using the conventions defined by Li et al. [1], assume that \( d \) is the maximum \( \maxflow(P_2) \) over all \( P_2 \) and the symbol \( \Omega \) represents a \( d \)-dimensional vector space over a large base field.

Let us define \textit{linear-code multicast (LCM)} \( \nu \) on a communication network \((\tilde{N}, P_1)\) with vector space \( \nu(P_\emptyset) \) assigned to every node \( P_2 \) and a vector \( \nu(P_\emptyset P_3) \) to every channel \( P_\emptyset P_3 \) such that

1) \( \nu(P_1) = \Omega \);

2) \( \nu(P_\emptyset P_3) \in \nu(P_\emptyset) \) for every channel \( P_\emptyset P_3 \);

3) For \( \emptyset \), collection of non-source nodes in the network

\[
\{ \nu(P_2) : P_2 \in \emptyset \} = \{ \nu(P_\emptyset P_3) : P_\emptyset \notin \emptyset, \ P_3 \in \emptyset \}. 
\]

Figure 4.2: Max-Flow for Network \( \tilde{N} \).

Figure 4.3: Max-Flow for Network \( M \).
Example 1: Suppose in network $\tilde{N}$ of figure 4.1(a), $P_1$ multicasts two bits $d_1$ and $d_2$ to destination nodes $P_3$ and $P_7$. Using LCM $\nu$ specified by

\[
\nu(p_1p_2) = \nu(p_2p_3) = \nu(p_2p_3) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}
\]

\[
\nu(p_1p_4) = \nu(p_4p_5) = \nu(p_4p_7) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}
\]

\[
\nu(p_5p_6) = \nu(p_6p_2) = \nu(p_6p_7) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}
\]

Example 2: Suppose in network $M$ of figure 4.1(b), $P_1$ multicasts two bits $d_1$ and $d_2$ to destination nodes $P_3$, $P_6$ and $P_7$. Using LCM $\nu$ specified by

\[
\nu(p_1p_2) = \nu(p_1p_4) = \nu(p_2a) = \nu(p_3b) = \nu(p_2p_3) = \nu(p_4p_7) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}
\]

\[
\nu(p_1p_5) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}
\]

\[
\nu(ac) = \nu(cP_3) = \nu(cP_6) = \nu(bd) = \nu(dp_6) = \nu(dp_7) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}
\]

Data sent on a channel is the product of information vector with the assigned channel vector. Example: data set on $P_5 P_6$ is $d_1 \oplus d_2$ in network $\tilde{N}$ and data set on $ac$ and $bd$ is $d_1 \oplus d_2$ in network $M$ of figure 4.1.

### 4.3 LCM–PA (Linear-Code Multicast on Parallel Architectures)

To demonstrate network coding potential for parallel networks and justify our approach, we have implemented it on Recursive Diagonal Torus (RDT), Multi Mesh of Trees (MMT), Mesh of Trees (MoT), Multi-Mesh (MM) and 2D-Mesh networks [116, 118, 119, 125, 126]. We have implemented the conventions of network coding to ensure that communication in these parallel networks can improve. For clarity, we have defined the topological properties of different parallel architectures. For diverse networks, different communication steps are used to show the flow of data in each step. For each network different algorithms are used and the data flow varies for each step. Table 3.1 shows characteristics of various processor organizations based on some of the network optimization parameters.
4.3.1 Network coding on RDT (Recursive Diagonal Torus)

RDT (figure 4.4) network provides better performance than 2D/3D/4D torus network and is developed with the best use of recursively structures diagonal torus connection [124]. This network has a diameter 11 for $2^{16}$ nodes and 8 edges per node. However, in other parallel networks (while communicating between different nodes having different data) the receiving node can receive one data from source nodes at a time (shown in figure 4.5). Due to arrival of multiple data at a receiver node it results data loss. This is handled by coding such nodes so that no data loss may occur. The faulty nodes are coded to XOR the data received from multiple nodes. This will eradicate the problem of data loss. In RDT topology, network coding principle is implemented to remove the faulty nodes and accomplish data communication. Figure 4.5 shows RDT topology which contains 32 communicating nodes out of which 8, 10-16, 18-24, 26-32 are faulty nodes. These nodes are receiving more than one input at unit time which eradicates either of the received data. This makes communication in this network complex. The communication is possible by using the network coding principle to XOR multiple data received at a node in unit time. Figure 4.6 shows an implementation of network coding in RDT network.

![Figure 4.4: RDT network.](image)

To implement network coding on RDT network, it is required to find maxflow to evaluate total number of inputs to different nodes. While calculating maxflow in figure 4.6, nodes receiving a different number of data are to be identified (table 4.1). The non-source nodes are marked for evaluating a different number of inputs to nodes. These non-source nodes are shown in figure 4.7.
Table 4.1: Maxflow in RDT network.

<table>
<thead>
<tr>
<th>Nodes Receiving Single Data Input</th>
<th>Nodes Receiving Two Data Inputs</th>
<th>Nodes Receiving Three Data Inputs</th>
<th>Nodes Receiving Four Data Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxflow (P_2) = 1</td>
<td>maxflow (P_9) = 2</td>
<td>maxflow (P_{16}) = 3</td>
<td>maxflow (P_{32}) = 4</td>
</tr>
<tr>
<td>maxflow (P_3) = 1</td>
<td>maxflow (P_{10}) = 2</td>
<td>maxflow (P_{22}) = 3</td>
<td></td>
</tr>
<tr>
<td>maxflow (P_4) = 1</td>
<td>maxflow (P_{11}) = 2</td>
<td>maxflow (P_{23}) = 3</td>
<td></td>
</tr>
<tr>
<td>maxflow (P_5) = 1</td>
<td>maxflow (P_{12}) = 2</td>
<td>maxflow (P_{26}) = 3</td>
<td></td>
</tr>
<tr>
<td>maxflow (P_6) = 1</td>
<td>maxflow (P_{13}) = 2</td>
<td>maxflow (P_{29}) = 3</td>
<td></td>
</tr>
<tr>
<td>maxflow (P_7) = 1</td>
<td>maxflow (P_{14}) = 2</td>
<td>maxflow (P_{27}) = 3</td>
<td></td>
</tr>
<tr>
<td>maxflow (P_8) = 1</td>
<td>maxflow (P_{15}) = 2</td>
<td>maxflow (P_{28}) = 3</td>
<td></td>
</tr>
<tr>
<td>maxflow (P_9) = 1</td>
<td>maxflow (P_{16}) = 2</td>
<td>maxflow (P_{29}) = 3</td>
<td></td>
</tr>
<tr>
<td>maxflow (P_{10}) = 1</td>
<td>maxflow (P_{17}) = 2</td>
<td>maxflow (P_{30}) = 3</td>
<td></td>
</tr>
</tbody>
</table>

In the figure given below, RDT network consists of four different maxflow on different nodes. If the statistics of this network is evaluated, eight nodes (\(P_2, P_3, P_4, P_5, P_6, P_7, P_9\), and \(P_{17}\)) have maxflow = 1; fourteen nodes (\(P_8, P_{10}, P_{11}, P_{12}, P_{13}, P_{14}, P_{15}, P_{18}, P_{19}, P_{20}, P_{21}, P_{22}, P_{23}\), and \(\ldots\)) have maxflow = 2; and the remaining nodes have maxflow = 3 or 4. **Figure 4.5:** Nodes in RDT network with faulty nodes.
$P_{23}$, and $P_{25}$) have maxflow = 2; eight nodes ($P_{16}$, $P_{24}$, $P_{26}$, $P_{27}$, $P_{28}$, $P_{29}$, $P_{30}$, and $P_{31}$) have maxflow = 3; and one node ($P_{32}$) have maxflow = 4.

Figure 4.6: RDT network with network coding. Source node 1 transfers data $d_1d_2$ to destination node 32.

Figure 4.7: RDT network with different values of maxflow at different nodes.
The statistics of *maxflow* in RDT network is: as *maxflow* increases the number of nodes decreases. That is, when *maxflow* = 1, number of nodes = 8; when *maxflow* = 2, number of nodes = 14; when *maxflow* = 3, number of nodes = 8; and when *maxflow* = 4, number of nodes = 1. This validates possibility to implement network coding on RDT. The simulation results of this prospect are given in figure 4.8.

![Figure 4.8: Different values of *maxflow* at different number of nodes.](image)

To achieve linear code multicast in RDT network, a vector space is assigned to every node and a vector to every channel to achieve parameters as in section 4.1. Assuming that the flow of data between two nodes \( (d_1, d_2) \) is multicast (figure 4.4). Some channels in this network contain either \( d_1 \) or \( d_2 \) and some contain \( d_1 \oplus d_2 \). Based on the above evaluated *maxflow*, the vector space is generated (figure 4.9) and using this vector space the linear code multicast is achieved. The vector space is shown below:

\[
\begin{align*}
\nu(P_1 P_9) &= \nu(P_1 P_{29}) = \nu(P_9 P_{10}) = \nu(P_9 P_{16}) = \nu(P_9 P_{17}) = \nu(P_{17} P_{18}) = \nu(P_{17} P_{24}) = \nu(P_{17} P_{25}) \\
&= \nu(P_{25} P_{26}) = \nu(P_{25} P_{32}) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
\end{align*}
\]

\[
\begin{align*}
\nu(P_{10} P_{11}) &= \nu(P_{10} P_{18}) = \nu(P_{11} P_{12}) = \nu(P_{11} P_{19}) = \nu(P_{12} P_{13}) = \nu(P_{12} P_{20}) = \nu(P_{13} P_{14}) = \nu(P_{13} P_{21}) \\
&= \nu(P_{14} P_{15}) = \nu(P_{14} P_{22}) = \nu(P_{15} P_{16}) = \nu(P_{15} P_{23}) = \nu(P_{16} P_{24}) = \nu(P_{18} P_{19}) = \nu(P_{18} P_{26}) = \nu(P_{19} P_{27}) = \nu(P_{20} P_{21}) = \nu(P_{20} P_{28}) = \nu(P_{21} P_{22}) = \nu(P_{21} P_{29}) = \nu(P_{22} P_{23}) = \nu(P_{22} P_{30})
\end{align*}
\]
=v(P_{23}P_{24}) = v(P_{23}P_{31}) = v(P_{21}P_{2}) = v(P_{1}P_{8}) = v(P_{2}P_{3}) = v(P_{2}P_{10}) = v(P_{2}P_{30}) = v(P_{3}P_{4})
= v(P_{3}P_{11}) = v(P_{3}P_{31}) = v(P_{4}P_{5}) = v(P_{4}P_{12}) = v(P_{4}P_{32}) = v(P_{5}P_{6}) = v(P_{5}P_{13}) = v(P_{5}P_{25})
= v(P_{6}P_{7}) = v(P_{6}P_{14}) = v(P_{6}P_{26}) = v(P_{7}P_{8}) = v(P_{7}P_{15}) = v(P_{7}P_{27}) = v(P_{8}P_{16}) = v(P_{8}P_{28})
= \binom{0}{1}

= v(P_{24}P_{22}) = v(P_{26}P_{27}) = v(P_{27}P_{28}) = v(P_{28}P_{29}) = v(P_{29}P_{30}) = v(P_{30}P_{31}) = v(P_{31}P_{32})
= \binom{1}{1}

Figure 4.9: Channels with different data bits (d_1, d_2 and d_1 \oplus d_2). In time domain x-axis captions 1, 2 and 3 represents d_1, d_2 and d_1 \oplus d_2.

Vector space generated in equations 1, 2 and 3 represent channels in RDT networks with data \(d_1, d_2\) and \(d_1 \oplus d_2\). Equation 1 contains 10 channels, which carry data \(d_1\). Similarly, equation 2 and 3 contains 22 and 32 channels, which carry data \(d_2\) and \(d_1 \oplus d_2\). After evaluation of vector space it is observed that network coding is required on those nodes with matrix principles. Equation 3 states data channels with XOR value \((d_1 \oplus d_2)\) of \(d_1\) and \(d_2\). So network coding is essential on nodes concerned in connectivity of channels as in equation 3. As equation 3 consists of 32 channels which involves 23 nodes \((P_8, P_{10}, P_{16}, P_{18}, P_{24},\) and \(P_{25}, P_{32}\)), so network coding approach is implemented on these nodes.
4.3.2 Network coding on MMT and MoT (Multi Mesh of Trees and Mesh of Trees)

In this section, we have implemented the network coding on MMT and MoT. Both these networks have identical interblock connectivity [116, 126] (figure 4.10), therefore implementation of network coding for interblock connectivity of these networks will remain same.

![Figure 4.10](image)

**Figure 4.10:** Figure in left shows MoT and one block of MMT network with 3×3 nodes. The figure in right shows MMT network with 3²×3² nodes.

To implement network coding, it is necessary to estimate maxflow in these networks. We consider different size of these networks to optimize our approach and study network coding with more results. We have analyzed network coding approach on MMT and MoT with network size 16 (4×4) and 36 (6×6). Let us assume that the network size of a block of MMT and entire MoT is 4×4, i.e. 16 nodes in a block of MMT and MoT topology. For maxflow, classification of nodes with the different number of incoming is mandatory. Figure 4.11 shows different maxflow in MMT and MoT network. We have considered two data bits \{d_1, d_2\} with node 1 which it multicast to other nodes in the network. Channels \{(1, 5) and (1, 9)\} carries data \(d_1\) and channels \{(1, 2) and (1, 3)\} carries data \(d_2\).

Now, data bits \(d_1, d_2\) are multicast to other nodes in the network. Assuming node 1 is the active node [120], in first step, data \(d_1\) is sent to nodes (5, 9) and data \(d_2\) is sent to node (2, 3). In second step, \(d_1\) data is communicated to nodes (6, 7, 10, 11 and 13) and \(d_2\) data is communicated to (4, 6, 7, 10 and 11). This illustrates that in second step of data
communication in MMT and MoT network, nodes (6, 7, 10, and 11) receives both \( d_1 \) and \( d_2 \) data from different source (5, 9) and (2, 3). It shows that, this problem can be solved using network coding. In other words, without network coding, it is impossible to multicast two bits \( d_1 \) and \( d_2 \) per unit time from source (4, 7) and (2, 3) to nodes (6, 7, 10 and 11) [1]. The maxflow in MMT and MoT is evaluated as in figure 4.12.

**Figure 4.11:** Figure shows MoT and one block of MMT network with 4×4 nodes. The flow of data in each step is shown with different line types. The table shows channel involved in communication in each step.

**Figure 4.12:** 4×4 MoT and one block of MMT network with different values of maxflow at different nodes.
In the above figure, MoT and one block of MMT network consists of two different \textit{maxflow} on different nodes. If the statistics of this network is evaluated, seven nodes (P_2, P_3, P_4, P_5, P_8, P_9 and P_{13}) have \textit{maxflow}= 1; eight nodes (P_6, P_7, P_{10}, P_{11}, P_{12}, P_{14}, P_{15} and P_{16}) have \textit{maxflow} = 2. The statistics of \textit{maxflow} in these networks is: as \textit{maxflow} increases the number of nodes decreases, which results in similar trend as in RDT network. That is, when \textit{maxflow} = 1, number of nodes = 7; when \textit{maxflow} = 2, number of nodes = 8. This verifies possibility for implementation of network coding on parallel networks. Figure 4.13 shows that if the number of nodes increases then the possibilities of network coded nodes also increase. Using this statics the number of nodes in a huge network can easily be estimated.

![Figure 4.13](image-url)  
**Figure 4.13:** Different values of \textit{maxflow} at different number of nodes in MoT and one block of MMT network for 4\times4.

![Figure 4.14](image-url)  
**Figure 4.14:** MOT and one block of MMT network with network coding. Source node 1 transfers data \(d_1d_2\) to destination node 16.
Figure 4.14 shows the manner of data transfer from a source node to destination in MoT and MMT network. It shows the nodes where network coding is implemented. Nodes (1, 2, 3, 4, 5, 9 and 13) do not require coding as they receive one input. However, nodes (6, 7, 8, 10 and 14) receive multiple data so these nodes are coded. Remaining nodes (11, 12 and 15) are non-output nodes so network coding is not required on these nodes. The source node 1, transfers 2 bits of data \( d_1, d_2 \) to destination node 16, which receive 1 bit data i.e. \( d_1 \oplus d_2 \). The received data is decoded to get the original information. Network coding has not only simplified the algorithm to transfer data but also reduced the size of data transfer to destination. After coding multiple data receiving nodes in MoT and MMT network, it can be formulated that network coding support to diminish the time complexity of multi-processing environments.

Further, only network coding can make sensible parallel processing. Let us reconsider the \( \text{maxflow} \) parameter for 6×6 MoT and one block of MMT for analyzing network coding more evidently on these networks. Assuming that the network size of a block of MMT and entire MoT is 6×6, i.e. 36 nodes in a block of MMT and MoT topology. \( \text{Maxflow} \) in the network is based on classification of nodes with respect to different number of incoming to each node. Figure 4.15 shows different \( \text{maxflow} \) in MMT and MoT network. We considered same data set, which consist of two data bits \( \{d_1, d_2\} \) with node 1, which it multicast to other nodes in the network. Channels \{ (1, 7) and (1, 13) \} carries data \( d_1 \) and channels \{ (1, 2) and (1, 3) \} carries data \( d_2 \).

![Figure 4.15: 6×6 MoT and one block of MMT network with different values of \( \text{maxflow} \) at different nodes.](image)
According to figure 4.15, MoT and one block of MMT network consists of two different maxflow on different nodes. From 36 nodes, ten nodes (\(P_2, P_3, P_4, P_5, P_6, P_{13}, P_{19}, P_{25}\) and \(P_{31}\)) have maxflow = 1; 25 nodes (\(P_8\sim P_{12}, P_14\sim P_{18}, P_{20}\sim P_{24}, P_{26}\sim P_{30}\) and \(P_{32}\sim P_{36}\)) have maxflow = 2. Now as maxflow increases, the number of nodes decreases which results in similar trend as in RDT network. That is, when maxflow = 1, number of nodes = 10; when maxflow = 2, number of nodes = 25. The simulation result is shown in figure 4.16. Results of maxflow on MoT and MMT are accessed above and vector space is generated. The vector space for MoT and MMT network is used to achieve linear code multicast. We are interested in the channels carrying XOR of data bits \((d_1, d_2)\) which is encoded using network coding. These channels are the source of complexity estimation in parallel networks. We have formulated the vector space generated using maxflow for both 4\(\times\)4 and 6\(\times\)6 network sizes.

\[
\begin{align*}
\nu(P_1P_3) &= \nu(P_1P_9) = \nu(P_5P_6) = \nu(P_5P_7) = \nu(P_9P_{10}) = \nu(P_9P_{11}) = \nu(P_{13}P_{14}) = \\
\nu(P_{13}P_{15}) &= \binom{1}{0} \\
\nu(P_1P_2) &= \nu(P_1P_3) = \nu(P_2P_4) = \nu(P_2P_6) = \nu(P_2P_{10}) = \nu(P_2P_{11}) = \nu(P_3P_{11}) = \nu(P_4P_8) = \\
\nu(P_4P_{12}) &= \binom{0}{1} \\
\nu(P_6P_{14}) &= \nu(P_7P_{15}) = \nu(P_8P_{16}) = \nu(P_{10}P_{12}) = \nu(P_{14}P_{16}) = \binom{1}{1}
\end{align*}
\]

**Figure 4.16:** Different values of maxflow at different number of nodes in MoT and one block of MMT network for 6\(\times\)6 size.
The above equations are the representation of vector space generated for MoT and one block MMT with network size 4×4. Equation 4 represents the channels which carry $d_1$ data to other nodes of network. In the same way, equation 5 represents channels with $d_2$ data. However, equation 6 consists of channels with XOR both $d_1$, $d_2$ i.e. $(d_1\oplus d_2)$. The vector space for 6×6 network size is shown below. Equation 9 consists of 30 channels which involve 25 nodes ($p_{8}$, $p_{12}$, $p_{14}$, $p_{18}$, $p_{20}$, $p_{24}$, $p_{26}$, $p_{30}$, and $p_{32}$, $p_{36}$), so network coding approach is implemented on these 25 nodes from both networks. Channels with different data bits ($d_1$, $d_2$ and $d_1\oplus d_2$) for both 4×4 and 6×6 network size is shown in figure 4.17.

\[
\nu(p_1p_7) = \nu(p_1p_{13}) = \nu(p_7p_9) = \nu(p_7p_{19}) = \nu(p_{13}p_{14}) = \nu(p_{13}p_{15}) = \nu(p_{13}p_{31}) = \nu(p_{19}p_{20}) = \nu(p_{19}p_{21}) = \nu(p_{25}p_{26}) = \nu(p_{25}p_{27}) = \nu(p_{31}p_{32}) = \nu(p_{31}p_{33}) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} 
\]  

\[
\nu(p_1p_2) = \nu(p_1p_3) = \nu(p_2p_4) = \nu(p_2p_5) = \nu(p_2p_8) = \nu(p_2p_{14}) = \nu(p_3p_6) = \nu(p_3p_9) = \nu(p_3p_{15}) = \nu(p_4p_{10}) = \nu(p_3p_{11}) = \nu(p_5p_{17}) = \nu(p_6p_{12}) = \nu(p_6p_{16}) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}
\]  

\[
\nu(p_8p_{10}) = \nu(p_8p_{11}) = \nu(p_8p_{20}) = \nu(p_9p_{26}) = \nu(p_9p_{21}) = \nu(p_9p_{27}) = \nu(p_{10}p_{22}) = \nu(p_{10}p_{28}) = \nu(p_{11}p_{23}) = \nu(p_{11}p_{29}) = \nu(p_{12}p_{24}) = \nu(p_{12}p_{30}) = \nu(p_{14}p_{16}) = \nu(p_{14}p_{17}) = \nu(p_{14}p_{32}) = \nu(p_{15}p_{18}) = \nu(p_{16}p_{34}) = \nu(p_{17}p_{35}) = \nu(p_{18}p_{36}) = \nu(p_{20}p_{22}) = \nu(p_{20}p_{23}) = \nu(p_{21}p_{24}) = \nu(p_{26}p_{28}) = \nu(p_{26}p_{29}) = \nu(p_{27}p_{30}) = \nu(p_{32}p_{34}) = \nu(p_{32}p_{35}) = \nu(p_{33}p_{36}) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}
\]

Figure 4.17: Channels with different data bits ($d_1$, $d_2$ and $d_1\oplus d_2$). In time domain x-axis captions 1, 2 and 3 represents $d_1$, $d_2$ and $d_1\oplus d_2$. Both 4×4 and 6×6 network size are represented above.
4.3.3 Network coding on Multi Mesh and 2D Mesh Network

A $n \times n$ multi mesh network [118] is a combination of $n^2(n \times n)$ 2D mesh [119]. We have considered both networks for implementing network coding. We have not considered the inter block connectivity between $n^2$ 2D mesh. Rather, we are using one 2D mesh to study the implementation results of network. A 2D mesh consists of $n^2$ nodes arranged in (row×column) format (figure 4.18). The diameter of this network is $(n-1)$ with a constant number of channels and channel length. We have considered $4 \times 4$ and $6 \times 6$ network size to study the variation in results after implementing network coding. Further we initiated with $4 \times 4$ network size to find the faulty nodes, as network coding is utmost required on these nodes. For this, assuming that node 1 ($P_1$) broadcast data $\{d_1, d_2\}$ to destination node 16 ($P_{16}$). Channel $\{1, 5\}$ carry $d_1$ and channel $\{1, 2\}$ carry $d_2$. Now, while communicating between different nodes with different data, the receiving nodes can receive either data from different input channels. Such different input receiving nodes are faulty nodes (figure 4.19). These nodes require network coding to eliminate this dilemma. In multi mesh and 2D mesh topology network coding principle is implemented to remove faulty nodes and accomplish data communication between nodes at unit time.

![2D Mesh and one block of Multi Mesh network](image)

**Figure 4.18:** 2D Mesh and one block of Multi Mesh network.

For implementing network coding on the 2D mesh and one block of multi mesh it is necessary to find maxflow in these networks. Maxflow is calculated based on number of incomings to a node. In figure 4.19 we have identified the faulty node with more than one incoming. These non-source nodes with different maxflow are shown in figure 4.20 and these are listed in table 4.2.
The above figure 4.20 shows different maxflow on these networks. The tendency of maxflow in these networks evaluates that six nodes (P_2, P_3, P_4, P_5, P_9 and P_{13}) have maxflow = 1; and nine node (P_6, P_7, P_8, P_{10}, P_{11}, P_{12}, P_{14}, P_{15} and P_{16}) have maxflow= 2. So, network coding is required for nodes having maxflow = 2, where more than one input is received. Calculation of
maxflow is required to identify nodes having a different number of input channels. The simulation result of this prospect is given in figure 4.21.

Figure 4.21: Different values of maxflow at different number of nodes in 2D Mesh and one block of Multi Mesh network for 4×4 size.

Figure 4.22: 2D Mesh and one block of Multi Mesh network with network coding. Source node 1 transfers data $d_1d_2$ to destination node 16.

Figure 4.22 shows the data transfer in the 2D mesh and multi mesh network from a source node to destination. Nodes (1, 2, 3, 4, 5, 9 and 13) do not require coding as they receive one input. However, nodes (6, 7, 8, 10, 11, 12, 14, 15 and 16) receive multiple data so these nodes are coded. The source node 1, transfers 2 bits of data ($d_1$, $d_2$) to destination node 16, which receive 1 bit data, i.e. $d_1 \oplus d_2$. The received data is decoded to get the original information.
Further we reconsidered the maxflow parameter for (6×6) 2D mesh and multi mesh network for examining network coding. Figure 4.23 shows different maxflow in 2D mesh and multi mesh network. We have considered same data set consisting of two data bits \( \{d_1, d_2\} \) with node 1 which it multicast to other nodes in the network.

The above figure illustrates that 2D mesh and one block of multi mesh network consist two different maxflow on different nodes. Out of 36 nodes, ten nodes \( (P_2, P_3, P_4, P_5, P_6, P_7, P_{13}, P_{19}, P_{25} \text{ and } P_{31}) \) have maxflow = 1; 25 nodes \( (P_8–P_{12}, P_{14}–P_{18}, P_{20}–P_{24}, P_{26}–P_{30} \text{ and } P_{32}–P_{36}) \) have maxflow = 2. Now, maxflow in these networks of 36 nodes increases, which results in similar trend as in RDT, MMT and MoT networks. That is, when maxflow = 1, number of nodes = 10; when maxflow = 2, number of nodes = 26. The simulation result of this observation on MoT and MMT is shown in figure 4.24.
Outcome of maxflow on the 2D mesh and one block of multi mesh is reviewed above and vector space is generated. The vector space is used to achieve linear code multicast on these networks. We are concerned about the channels which carry an XOR of data bits $d_1, d_2$ which necessitate network coding. These channels are the source of complexity estimation in parallel networks. Now we formulate the vector space generated using maxflow in these networks for both 4×4 and 6×6 network size.

$$v(P_1P_3) = v(P_3P_6) = v(P_3P_9) = v(P_9P_{10}) = v(P_{13}P_{14}) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$ \hfill (10)

$$v(P_1P_2) = v(P_2P_6) = v(P_2P_3) = v(P_3P_4) = v(P_3P_7) = v(P_4P_8) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$ \hfill (11)

$$v(P_6P_7) = v(P_6P_{10}) = v(P_7P_8) = v(P_7P_{11}) = v(P_8P_{12}) = v(P_{10}P_{11}) = v(P_{10}P_{14}) = v(P_{11}P_{12}) = v(P_{11}P_{15}) = v(P_{12}P_{16}) = v(P_{14}P_{15}) = v(P_{12}P_{16}) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$ \hfill (12)

The above equations are representations of vector space generated for 2D mesh and one block of multi mesh with network size 4×4. Equation 10 represents the channels which carry $d_1$ data and equation 11 represent channels with $d_2$ data to other nodes of network. Equation 12 represents channels with XOR of both $d_1, d_2$ i.e. $(d_1 \oplus d_2)$. The vector space for 6×6 network
size is shown below. Equation 15 consists of 40 channels which involve 25 nodes \((P_8, P_{12}, P_{14}, P_{18}, P_{20}, P_{24}, P_{26}, P_{30}, \text{and } P_{32})\), so network coding approach is implemented on these 25 nodes. Channels with different data bits for both \((4 \times 4)\) and \((6 \times 6)\) network size are represented in figure 4.25.

\[
\begin{align*}
\nu(P_1P_7) &= \nu(P_7P_8) = \nu(P_1P_{13}) = \nu(P_{13}P_{14}) = \nu(P_{19}P_{20}) = \nu(P_{19}P_{25}) = \nu(P_{25}P_{26}) = \\
\nu(P_{25}P_{31}) &= \nu(P_{31}P_{32}) = \left(\begin{array}{c} 1 \\ 0 \end{array}\right) \\
\nu(P_1P_2) &= \nu(P_2P_8) = \nu(P_3P_4) = \nu(P_3P_9) = \nu(P_4P_{10}) = \nu(P_5P_6) = \nu(P_3P_{11}) = \\
\nu(P_6P_{12}) &= \left(\begin{array}{c} 0 \\ 1 \end{array}\right) \\
\nu(P_3P_8) &= \nu(P_8P_{14}) = \nu(P_9P_{10}) = \nu(P_9P_{15}) = \nu(P_{16}P_{11}) = \nu(P_{10}P_{16}) = \nu(P_{11}P_{12}) = \nu(P_{11}P_{17}) = \\
\nu(P_{12}P_{18}) &= \nu(P_{14}P_{15}) = \nu(P_{14}P_{20}) = \nu(P_{15}P_{16}) = \nu(P_{15}P_{21}) = \nu(P_{16}P_{17}) = \nu(P_{16}P_{22}) = \\
\nu(P_{17}P_{18}) &= \nu(P_{17}P_{23}) = \nu(P_{18}P_{24}) = \nu(P_{20}P_{21}) = \nu(P_{20}P_{26}) = \nu(P_{21}P_{22}) = \nu(P_{22}P_{27}) = \\
\nu(P_{22}P_{23}) &= \nu(P_{22}P_{28}) = \nu(P_{23}P_{24}) = \nu(P_{23}P_{29}) = \nu(P_{24}P_{30}) = \nu(P_{25}P_{27}) = \nu(P_{26}P_{32}) = \\
\nu(P_{27}P_{28}) &= \nu(P_{27}P_{33}) = \nu(P_{28}P_{29}) = \nu(P_{28}P_{34}) = \nu(P_{29}P_{30}) = \nu(P_{29}P_{35}) = \nu(P_{30}P_{36}) = \nu(P_{32}P_{33}) = \\
\nu(P_{33}P_{34}) &= \nu(P_{34}P_{35}) = \nu(P_{35}P_{36}) = \left(\begin{array}{c} 1 \\ 1 \end{array}\right)
\end{align*}
\] — (15)

Figure 4.25: Channels with different data bits \((d_1, d_2, \text{and } d_1 \oplus d_2)\). In time domain x-axis captions 1, 2 and 3 represents \(d_1, d_2, \text{and } d_1 \oplus d_2\). Both \((4 \times 4)\) and \((6 \times 6)\) 2D Mesh and one block of MMT network size are represented above.
4.4 Benefits of Linear Network Coding on Parallel Networks

Network coding suggests considerable capacity gains for networks with special structures [127]. Advantages offered by network coding for different network scenarios are different. For parallel architectures, network coding offers some generic solutions to a set of problems. In this section, we considered three generic solutions to the above specified parallel architectures (Recursive Diagonal Torus (RDT), Multi Mesh of Trees (MMT), Mesh of Trees (MoT), Multi Mesh (MM) and 2D-Mesh networks) in which linear coding is particularly useful.

4.4.1 Removal of Faulty Nodes

In parallel networks, large numbers of nodes are involved and all nodes act as sender and receiver at different part of communication. Several nodes in this communication may receive diverse data from different nodes. However, a node can only receive one bit data at a particular time. So, either the information sent from multiple sources is lost or incomplete information is received by the node. Consider the problem of sending two bits of data from a source node to destination node in RDT network (figure 4.5). For simplicity, we analyzed only one node (P10) which receive two diverse data (d1 and d2) from two channels. Now, either d1 or d2 can be received or complete information is lost. This node is now acting as a faulty node. This setback can only be removed by using network coding. When network coding is implemented this faulty node will XOR the data (d1 and d2) to (d1\(\oplus\)d2). So the chance of information failure at this node is reduced. Now this node will further transfer XOR data to other connected nodes and vice versa. To identify the faulty nodes in a parallel network, maxflow is calculated.

Evaluation of maxflow not only identifies faulty nodes but also analyze number of incomings to this node. Based on the level of maxflow, network coding is implemented. According to previous section in these parallel networks, network coding is suggested based on the levels of maxflow evaluated. While analyzing maxflow a generic trend is originated. This trend suggests that as the number of node increases, maxflow decreases. Figure 4.26 shows the trend in these parallel networks.
4.4.2 Reduced information size

Another scenario in which network coding can be advantageous, is reduced information size. We have compared communication without and with network coding, in which the size of information increases based on the input received by the nodes. In parallel networks a core criterion is, to reduce the size of information flow so that communication complexity is reduced. This setback makes practical implementation of parallel networks unfeasible.

![Figure 4.26: Trend of maxflow in above parallel networks (Recursive Diagonal Torus (RDT), Multi Mesh of Trees (MMT), Mesh of Trees (MoT), Multi Mesh (MM) and 2D-Mesh networks).](image)

We have applied network coding with random communication with the following parameters: number of nodes \( n = 4 \), number of source = 1, number of receiver = \((n \times n) - 1\), number of destination = 1, steps of communication involved = 6. To distinct approach with and without network coding, assume that each node consist of different one bit data size. Let us examine communication in this network using both approaches:

### 4.4.2.1 Communication without network coding

In first step, the source node transmits data to the immediate connected node (see figure 4.27). The data size at receiving nodes will become two bits each. These nodes will act as sender node for step 2. In step 2, two receiving node get data of size three bits i.e., (2+1) and
one get data size five i.e., (2+2+1). Similarly, till step 5 the data size will become thirty four bits of each node. The destination node receives data of size sixty eight with its previous data of size one bit; combined buffer size of this node is sixty nine. The simulation based on the size of data in each step without network coding is shown in figure 4.28.

Figure 4.27: Steps of communication without network coding.

Figure 4.28: Communication in 2D mesh network without network coding.

Figure 4.27 shows that without using the network coding in 2D mesh the data size in each step grows exponentially. This is a limitation of parallel architecture, which can be reduced using network coding. These simulations do not attempt to quantify accurately the differences
in performance and overhead of linear network coding with parallel networks, but are useful as a preliminary suggestion.

### 4.4.2.2 Communication with network coding.

We have considered above example of 2D mesh architecture and evaluated results by implementing network coding approach. The parameters and constraints are assumed similar as stated above. The steps of communication in the 2D mesh network will also remain unchanged. So the communication involves six steps from source to destination (figure 4.29).

The source node transfers data to immediate connected nodes, the receiving node encodes the data received. Now this XOR data is one bit size and is transferred further to other nodes. Similar encoding is performed at other nodes in step 2. This process is performed until the complete information is received at destination node which will finally hold one bit information in the buffer of the destination nodes.

Irrespective of time complexity involved at each node to encode data, the size of data will remain same. Due to encoding at each node the data is XOR to form single information. This will eliminate the problem of excess data size associated with parallel architectures at each node buffer. Network coding provides an alternative to reduce buffer size of each node in parallel networks. While communication in 2D mesh network with network coding, the data size remains constant. This is a revolutionary advantage in the field of parallel communication, which suggests a better technique to implement high-scale parallel architecture.

![Figure 4.29: Steps of communication in 2D mesh network with network coding.](image-url)
4.4.3 *Reduced algorithmic time complexity*

In previous part of this section, we have shown that using network coding; the buffer size of each node in a parallel topology can be reduced exponentially.

\[
\text{Time Complexity Variation for Different Network Size (x-axis) with and without network coding.}
\]

Now, due to this reduced data size the complexity of data communication between different nodes is reduced. We have considered the previous example of 2D mesh network (figure 4.27), where the communication is accomplished in six steps. Let us assume that each data element is of size \( n \). After step 1, time involved to communicate data is \( logn \). The complexity of communication after step 2 is \( nlogn \). Similarly, after step 6 the time complexity will become \( n^5logn \). This is the complexity to communicate data from source to destination in six steps. Now, by using network coding it is reduced to \( 6logn \). At each step the data size remains constant (\( logn \)) so it is \( 6logn \) after entire communication. This shows a vast difference in terms of time complexity in parallel networks (shown in figure 4.30). This approach resolves three inter-linked problems of parallel communication.
4.5 Chapter Outline

This chapter presents linear network coding on parallel architecture. This multisource multicast parallel networks approach shows that: faulty nodes, information size and time complexity of communication decreases with code length. Further this approach also achieves capacity asymptotically as given by max-flow min-cut bound. As a result a generic coding methodology for different parallel networks is proposed. By means of different network scenarios it is shown that linear network coding effectively reduces the probability of errors. Finally, it is shown that LCM–PA benefits over routing approaches and the robustness of linear network coding offers a significant advantage in practical exploitation of parallel architectures.