Instead of pointwise convolution we want a multi-layer network for fusing information among channels which has the following properties:

1. Full connectivity from every input to all outputs
2. Large information bottleneck
3. Low operation count
4. Operation symmetry

Butterfly Transform (BFT) has all the fusing network design principles with \( n \log(n) \) edges which holds the minimum. BFT splits the data to \( K \) parts. Connects \( i \)'th element of each part to \( i \)'th element of other parts in the first layer and recursively fuses information inside each part in the next layers. The figure shows the case when \( K = 2 \).

### Design Choices

- **(a)** Use very small weight decay
- **(b)** Don't use non-linearity inside BFT
- **(c)** Use Butterfly Base \( K = 4 \)
- **(d)** Add residual connections

### Table: Residual Connections

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>NonResidual</td>
<td>79.1</td>
</tr>
<tr>
<td>Every-Other Layer</td>
<td>84.12</td>
</tr>
<tr>
<td>Post-to-Last</td>
<td>87.75</td>
</tr>
</tbody>
</table>

In BFT we should not enforce weight decay because it significantly reduces the effect of input channels on output channels. Similarly, we should not apply the common non-linear activation functions. These functions zero out almost half of the values in the intermediate BFLayers, which leads to a catastrophic drop in the information flow from input channels to the output channels. Butterfly base determines the structure of BFT. Under 60 M FLOP budget base \( K = 4 \) works the best. Each BFT adds \( \log(n) \) layers to the network. Residual connections generally help when training very deep networks.