Self-Adjusting Double Failure Tolerating Disk Arrays

Visualization with \( k \) regular graphs

Thomas J. E. Schwarz, S.J., Jehan-François Pâris, Darrell D. E. Long, Ethan L. Miller

Santa Clara University, tjschwarz@scu.edu
University of Houston, paris@acm.org
University of California at Santa Cruz, darrell@cse.ucsc.edu
Two Failure Tolerant Disk Arrays

- Disk drives fail as devices.
  - Rate can be as high as 5-10% per year
- Disk suffer block corruption
  - Latent error, often detected in RAID when trying to reconstruct
  - Can use scrubbing to detect
- Push towards two failure tolerant disk arrays
Two Failure Tolerant Disk Arrays

- Use coding to provide two failure tolerance
  - Reed Solomon Codes
    - Most flexible, most computationally expensive
      - Though actually not as expensive as is claimed
  - Use XOR based codes
    - Row diagonal parity
    - Even Odd
    - X codes
    - Park’s code
    - ...
  - Place every data disk in two disjoint reliability stripes
Two Failure Tolerant Disk Arrays

- Hellerstein, Gibson, et al.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>a</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>b</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>c</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>d</td>
</tr>
<tr>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td></td>
</tr>
</tbody>
</table>

Two Failure Tolerant Disk Arrays

- Each data disk is in exactly $r$ reliability stripes.
- Each reliability stripe contains exactly $k$ data disks.
- Two different data disks are in at most one reliability stripe.

**Proposition** [Gr96]: The necessary conditions for the existence of a configuration with parameters $v, r, b, k$ are:

$$vr = bk$$

$$v \geq r(k - 1) + 1.$$
Graph Representation

Dual of a two failure tolerant configuration is a $k$-regular graph
Graph Representation

- Can also be obtained from generator matrix of code
- Parity matrix is the vertex-edge matrix of graph

\[
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1
\end{pmatrix}
\]
Graph Representation

- Allows us to quickly construct a vast number of 2 failure tolerant disk array layouts
Data Reconstruction

- Failed parity disk is a missing vertex
  - Can immediately reconstruct if all edges are still there
- Failed data disk is a missing edge
  - Can immediately reconstruct if all other edges at one of the vertices are still there.
Data Reconstruction

- Failure pattern visualized as loss of edges and vertices

*Proposition 1:* A failure pattern leads to data loss if it contains (a) a cycle of edges or (b) an path and both of its endpoints.

Reconfiguration

- Medium, big disk arrays will have spares
- After spares are used up, reconfigure array
  - Performance goes down
  - 2-failure tolerance is maintained
Reconfiguration
Reconfiguration
Reconfiguration

- Further reconfiguration is impossible
- Graph is no longer 4-regular
  - Parity drive load is increased
Reconfiguration

**Proposition 2:** The number $f$ of reconfigurations of a two failure tolerant disk array with initially $b$ parity disks and $v$ data disks and $k$ data disks per reliability stripe is at most
Better = Tighter Codes

- Use virtual data, parity disks
- Place several on the same storage device
- 2-failure tolerance gets new meaning
  - failures of virtual disks
  - failures of storage devices
- 2-failure tolerant remains if
  - e.g. virtual disks are at a distance $\geq 3$ in graph
  - use backtracking to find assignments on the fly
Better = Tighter Codes

- Reorganization still operates as before
- Future work:
  - Develop algorithms for layout
  - Take ease of data reconstruction into account